California Institute of Technology

Computer Science Department

5016:TR:82

BRISTLE BLOCKS - SCRUTINIZED AND ANALYZED

by

Richard McNair,
Xerox' Representative to SSP

and
Monroe Miller
Bendix' Representative to SSP

June 1982

Introduction

The primary purpose of this document is to record the artwork tracing and reduction to circuit diagrams of an entire chip design generated by Bristle Blocks. Bristle Blocks is a "Silicon Compiler" developed at California Institute of Technology (Ref. BB). The major design documented is a simple storage chip with an I/O port to get data in and out. Many of the circuits traced are common to all Bristle Block designs and may therefore be useful as a background for detailed looks at more complex Bristle Block designs. Another reason to document this chip is that an eight bit version is being fabricated to test the ability of a simple Bristle Blocks chip to actually function. Several other reasons to document this chip are to describe its successful logic simulation with MOSSIM II (Ref. MSII) and to describe some unexpected ways in which the I/O Port operated. It is also felt that there is some tutorial value in seeing the operation of an entire (simple) system with two phase clocking - - as is exemplified by this chip.

The chip was named REG2bt for the purposes of a six character (maximum) name as required by Bristle Blocks. The chip will be referred to as REG2bt.

This document will also comment briefly about another Bristle Block designs in terms of control equation implementation and related "surprises", circuit schematic and simulation with MOSSIM II.

REG2bt Design Description

The entire chip design was described to Bristle Blocks ***(hereafter referred to as BB)*** with the following twenty simple lines (plus blank lines) of text.

```
name      reg2bt  2; "Monroe's register test down to 2 bits"

field     clear<1>,load<2>,op<3>;

register register
   options:
     [read-upper: load=i,
     write-upper: op=i,
     refresh: always,
     suggest: clear=i,
     value:  oo];

precharge-both bus-charger;

io-port      io-port
   output-register:
     [write-upper: load=i,
     refresh load=o],
   load load=i,

   input-register:
     [read-upper op=i,
     refresh always],
   drive op=i;

end
```

Comments:
Data Path width is 2 bits per 1st line

Three control signals exist per 2nd line (two other control signals are generated per the definition of the `"io-port"`*)*

"True" is indicated by `i`*, "false" by* `o`*, e.g.*
`load=i` *means when load is true*
`load=o` *means when load is false*

In general, element controls may involve logic equations ("and"/"or"/"not" etc.) of control signals. This example is simplistic and has no logic operations.

The BB document [ref. BB] is the source for the complete explanation the design statement. However, only by tracing the artwork was it possible to determine how to properly set up the IO-PORT. It is believed that proper use of the IO-PORT to get artwork that can operate properly needs to be described because the BB manual examples appear wrong.

The first (minor) surprise was that two control signals requiring proper pad inputs were generated by the IO-PORT. The second (major) surprise was that the IO-PORT's register named "INPUT REGISTER" seemed to function as an output register and vice versa for the "OUTPUT REGISTER." The nature of the appropriate signals for all the IO-PORT's pads will be described after the composite schematic for the entire chip has been developed.

The reader may find it interesting to look at the seven artwork figures further back in this document in order to get a perspective on how much chip design is created from such a small design statement. The "high level" compiler language did indeed produce a great deal of "low level" detail. This document builds up information so that the reader may see some direct relationships between the design statement and the top of figure RegTot.02.sil.

Appendix A provides a list of all the data path elements allowed by BB. Many readers may get some idea of the power intended for a silicon compiler by simply seeing the list of elements allowed. REG2bt used the elements whose "description order numbers" (left margin of App. A) are 1, 24 and 14.

Documentation Conventions Used Herein

BB uses the underscore character available on the DEC 20 (the computer on which BB operates). That character translates to back arrow on many other computers (such as the one used to generate this document). Therefore, this document will use a hyphen where BB used the underscore. For example:

```
IO-PORT rather than IO←PORT or IO_PORT.
```

The artwork generated by BB is in CIF format. The CIF file was transmitted to Xerox and the artwork displays in this document were created by the Xerox Cell Design System - CDS [ref. CDS]. (CDS is not available outside Xerox.) Lines and text (including Q# to identify transistors) were added. The stipple patterns associated with various layers follow the conventions shown on a following page entitled 'Stipple Patterns'.

The circuit diagrams in this document were created with the Xerox SIL system [ref. SIL]. (SIL is not available outside Xerox.) The conventions employed in the schematics are explained in a following page entitled 'Schematic Conventions'.

Circuit Tracing

The REG2bt circuits are traced by the combination of three types of figures. The first type is the layout plots on which circuit names are given, a few signals carried by the artwork layers are identified and, most importantly, individual transistors are identified. Figure "Artwork: page 1" gives an over view and "Artwork: pages 2 thru 7" get to the real detail.

The second type of figures is the schematics identified as REG01.sil through REG03.sil. In these schematics, the individual circuits are detailed by referencing the transistors in the artwork pages and some simplified logic symbols are identified for the major circuit functions.

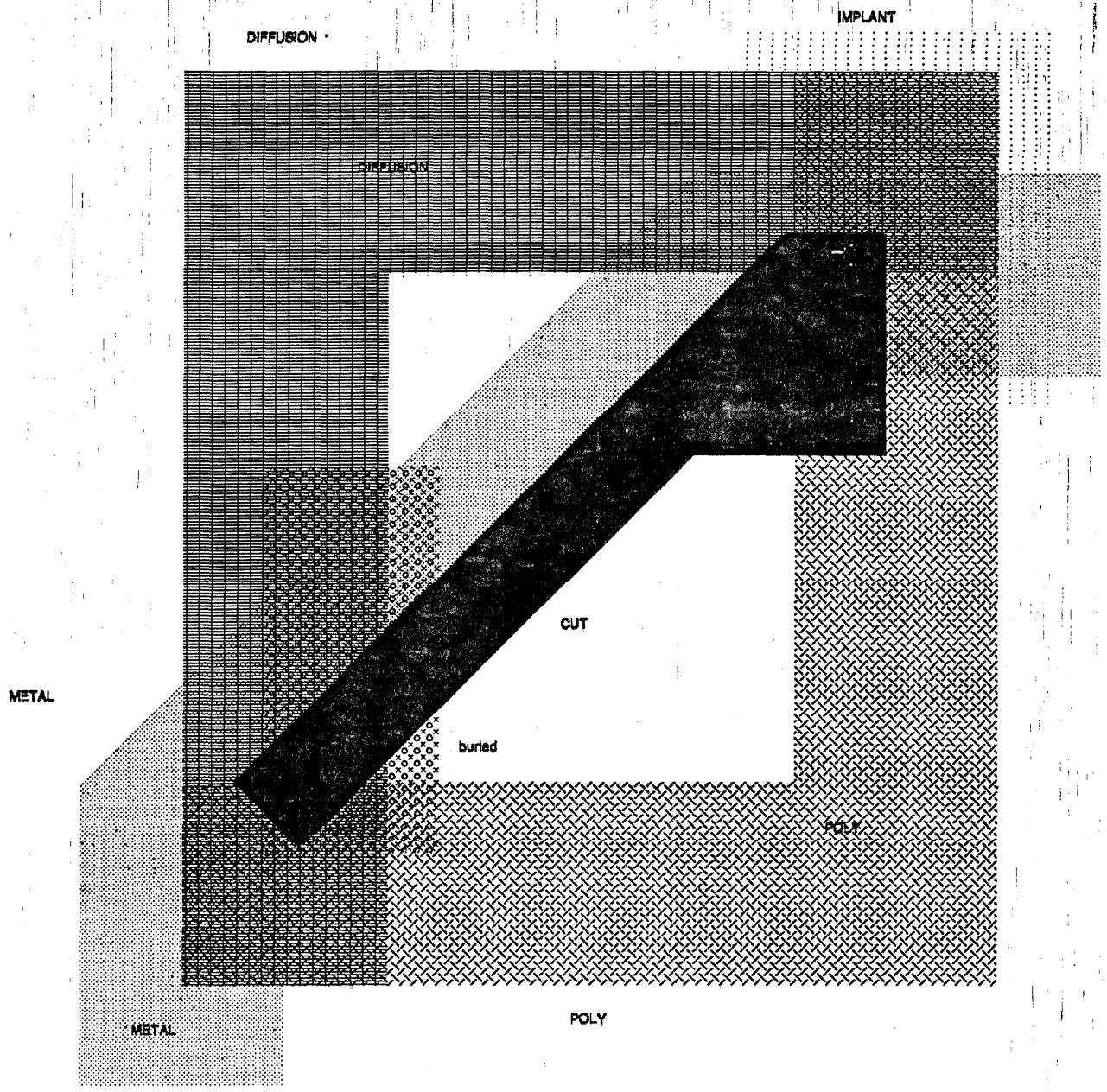
The third type of figures is the schematics identified as REGtot.01.sil and REGtot.02.sil. These

schematics are developed using the simplified logic symbols and the functionality of the entire chip is shown (all 292 transistors).

Before any of these types of figures are presented, there is a Figure entitled REG2bt Over View that presents a sketch of the chip floor plan. It shows the pads, assigns numbers used in subsequent documentation and identifies signal names for the pads. It also identifies the transistor content of the entire chip.

Some circuit tracing commentary will be provided before the artwork figures and the circuit schematics. Then, following the individual circuit schematics, some commentary will be provided regarding the composite chip schematic.

The REG2bt discussion concludes with a brief chip functional description and a presentation of the chip's successful MOSSIM II simulation. A final section provides a very brief look at a more complex BB design, with emphasis regards control equations in BB.



File: stip-1.press	Scale: 1, 1	User: monair.es	Date: 13-Apr-82 17:23:45	Page 1 of 1 [X: 1, Y: 1]

Stipple Patterns

TRANSISTORS

Depletion Transistors have a wide line as seen at left
Depletion Transistor = Implanted Transistor

Enhancement Transistors have a normal width line as seen at left

Transistors are identified with a number constructed as follows:

PageNumber.CircuitNumber.TransistorNumber

Example: 1.1.17

This is the ID for a transistor on page one, in circuit one, and it is transistor seventeen (17)

See examples below

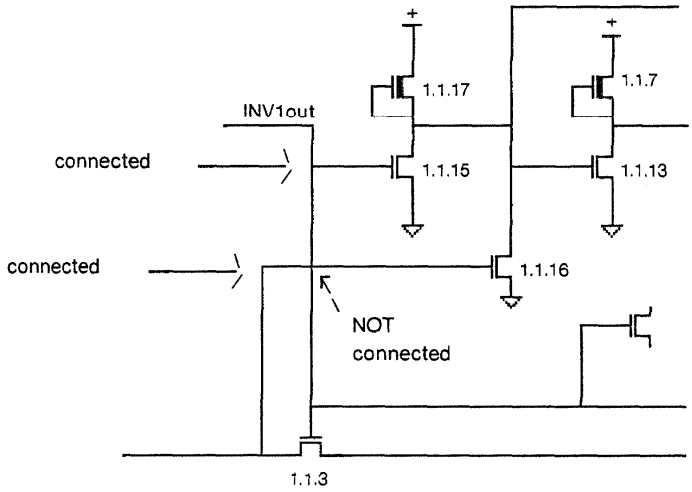
CONNECTIVITY

Lines that cross are NOT connected

Lines at a L shape are connected

Lines that terminate into another line are connected

See examples below



Conventions for Other Processing

For processing to network lists (not included in this report), there were several other conventions followed:

Signal names above horizontal lines and to the left of vertical lines

Comment text in [ ]

Logic symbols in boxes with italisized names for symbol and interface pin names near boarder box

PADS

Pad type numbers are shown inside boxes below

1	*Signal, Diffusion Res. Out*	1 Enh. Q,	6ea
2	Clock, Metal out	1 Enh. Q	4ea
3	Output pad w/ driver	6 Enh. Q, 4 Dep. Q	1ea
4	I/O pad w/ controller	12 Enh. Q, 5 Dep. Q	2ea
5	Power Pad	0 transistors	2ea

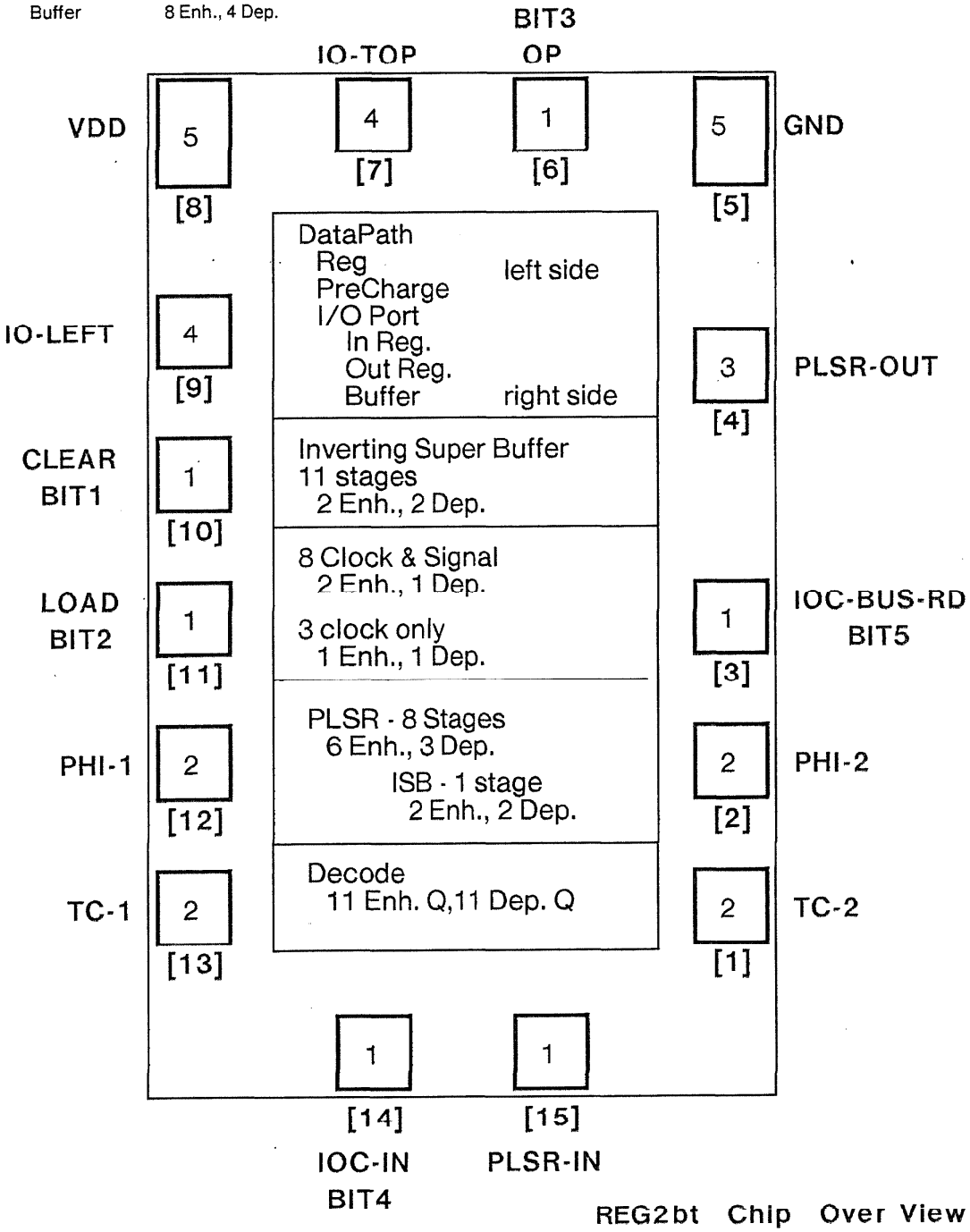
DataPath — Two each

Reg	6 Enh., 2 Dep.
PreCharge	2 Enh.
I/O Port	
In Reg.	4 Enh., 2 Dep.
Out Reg.	4 Enh., 2 Dep.
Buffer	8 Enh., 4 Dep.

General Notes:

Q is used as an abbreviation for transistor(s)

At times, just Enh. or Dep. is used as an abbreviation for transistor(s)



REG2bt Chip Over View

regoverview.sil

The "Artwork pages" present the segments of circuitry in the approximate order of signal flow from the pads to the datapath and out to the pads again.

"Artwork: page 1" is the only scale at which the simple input pads are shown. The Power pads (top corners) have no transistors associated with them and they directly route the power from the pads using interdigitated metal. No schematics are provided. There are transistors (of sorts - really they are protection circuits) associated with both the basic signal input pads and the clock input pads. Schematics are provided as circuits 3 and 4 in figure REG01.sil. The major reason to cover them here is that a circuit extraction program will identify the transistors just as is done in the schematics.

Several comments should be made before detailed artwork discussion. The technology output by BB is nMOS. Lambda is 3 microns (minimum transistor size is 6 microns). Butting (not buried) contacts are used. Depletion transistors allow the butting contacts to be over the active transistor area as is seen throughout the artwork (per acceptance of trade-offs discussed in Ref. M&C, page 44). Typical poly to diffusion butting contacts are seen in the top and bottom of the ISB (right side) of Artwork: page 3. In most instances, it is very difficult to confirm the presence or absence of a implant stipple pattern in the artwork figures (sorry about that).

"Artwork: page 2" presents a portion of the DECODE circuitry. In this simple REG2bt example, there are no logical "and" or "or" conditions associated with the control signals and the decoder therefore has depletion loads pulled down by only one enhancement device. The result is that the circuitry looks like simple inverters. In the more typical BB case, there would be "and's" or "or's" used with the control signals and the pull-ups would have multiple pull-downs in a Weinberger Nor Gate structure (see final section of this document for an example). The devices identified in the artwork trace for BIT1 (CLEAR) are shown as circuit 6 in figure REG03.sil. It is seen that the output signal is not inverted because two inverters are put in series.

"Artwork: page 3" presents a Parallel Load Shift Register (*PLSR*) that passes the signal through but also provides for test signals to be used to monitor the control operation per the documentation in the BB manual. This stage also provides for a first level of clocking. The devices identified in the artwork trace are shown as circuit 5 in figure REG03.sil.

"Artwork: page 3" also presents an Inverting Super Buffer (*ISB*) (Ref. M&C, page 17) that takes the final test output signal toward the pad from the Parallel Load Shift Register. The devices identified in the artwork trace are shown as circuit 1 in figure REG03.sil.

"Artwork: page 7" presents the Output Pad Driver (driven from the *ISB* described just above) that finishes taking the final test output signal from the Parallel Load Shift Register to the output pad. The devices identified in the artwork trace are shown as circuit 2 in figure REG01.sil.

The above two paragraphs discuss the test flow of the PLSR. Let us return to considering the major signal flow. The signals from the PLSR now go into the CLOCK-SIGNAL circuits seen at the bottom of "Artwork: page 4". This circuitry provides a second level of clocking. The devices identified in the artwork trace are shown as circuit 2 in figure REG03.sil.

This same level of circuitry also passes on "clock only" signals for use in the datapath. The right side of the lower boxed circuit ("Artwork: page 4") shows a stage with only clocks being generated. The left side's transistor identifiers are used as circuit 3 in figure REG03.sil, but it is

noted that transistor one does not exist.

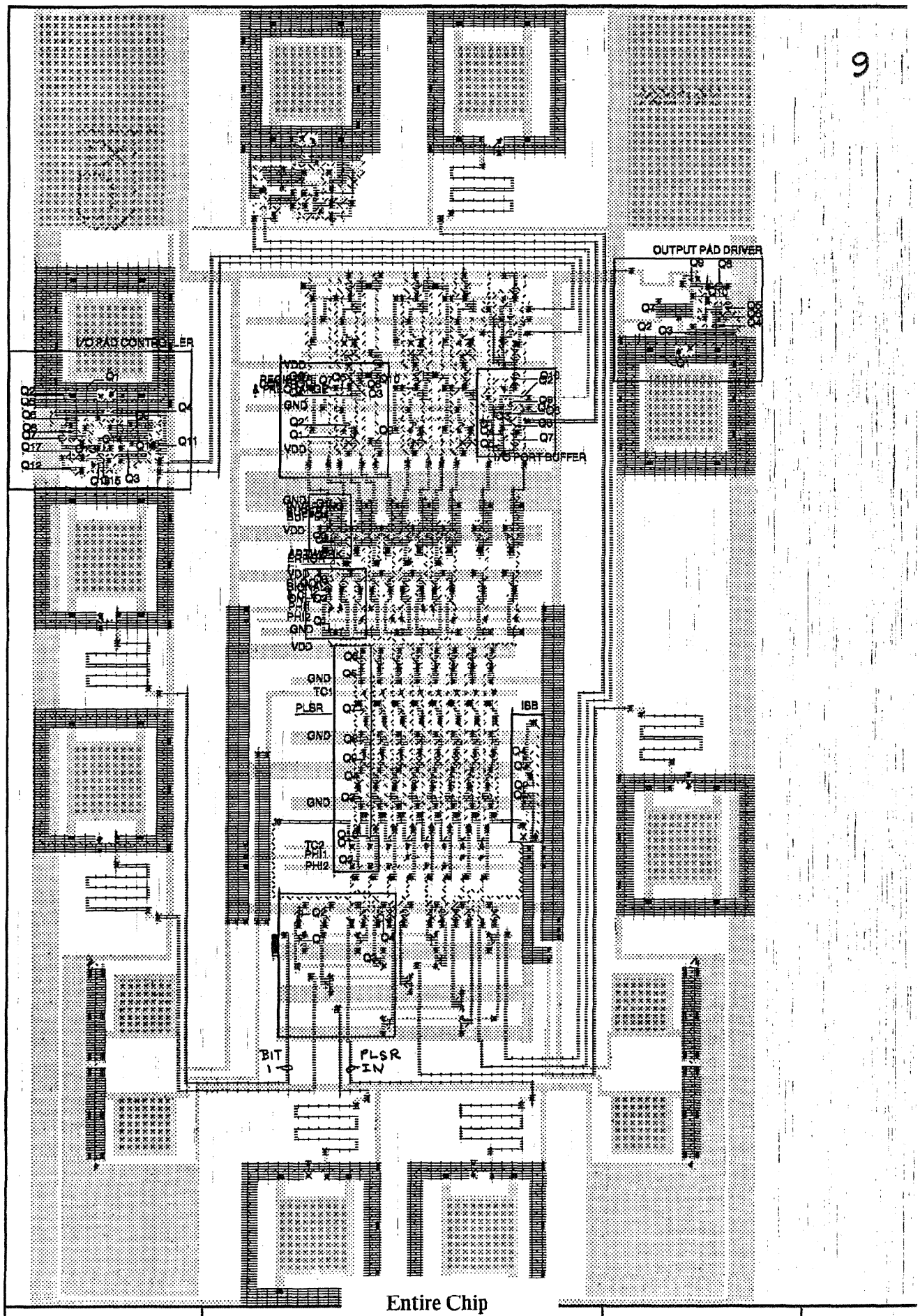
"Artwork: page 4" also presents an Inverting Super Buffer (*ISB*) that takes the clocked control signals and "clocks only" to the datapath. The devices identified in the artwork trace are shown as circuit 1 in figure REG03.sil. It is noted that the artwork figure has identified an error in which a poly line overlaps a diffussion line rather than being spaced one lambda away. This artwork feature had to be edited in the CIF file to be narrower in order for the circuit extractor to identify the transistors as functionally intended. Various experts have suggested that the circuit would function as shown with some possible degradation in performance or yield.

"Artwork: page 5" (bottom portion) gets us to the heart of the chip - the datapath. It presents the Register, Control & PreCharger. The devices identified in the artwork trace are shown as circuit 2 in figure REG02.sil. The Read/Write/Set pass transistors vary from one register to another, but the basic register is repeated three times as seen in the composite schematic, RegTot.02.sil. Therefore, a fundamental Pseudo Static Register (*PSR*) is defined as a basic logic symbol and the composite diagram uses this *PSR* and directly shows the external pass transistors. This same artwork/schematic set also shows the bus precharger.

"Artwork: page 5" also presents the I/O PORT Buffer (*IOB*) that takes the I/O Ports's output signal toward the pad. The devices identified in the artwork trace are shown as circuit 1 in figure REG02.sil.

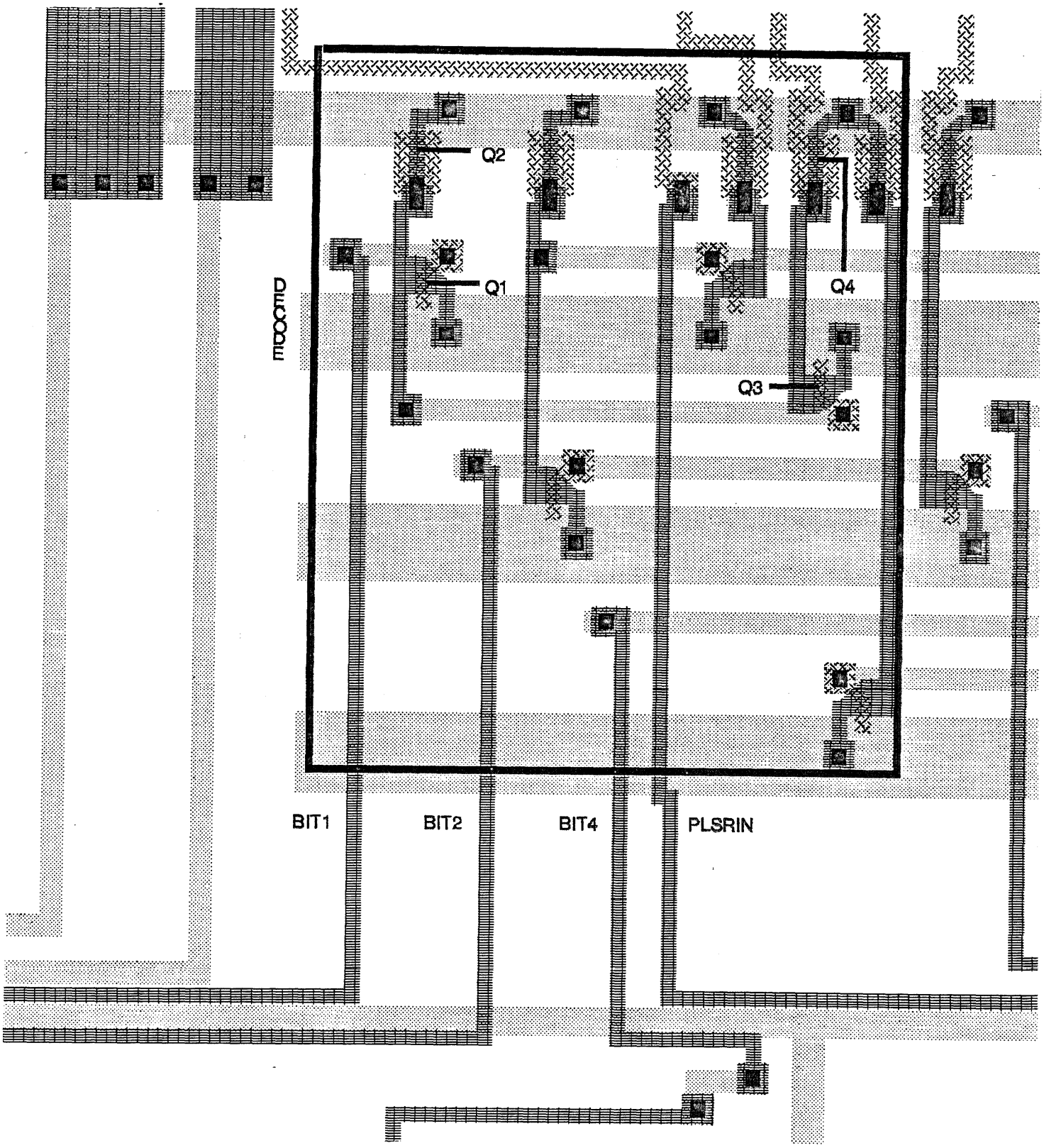
"Artwork: page 6" presents the I/O Pad controller (driven from the *IOB* described just above) that finishes taking the output signal to the pad. The devices identified in the artwork trace are shown as circuit 1 in figure REG01.sil.

The composite schematic to be presented in the next section will allow the individual circuits to be understood in a larger context that makes the design, especially the datapath, more clear.

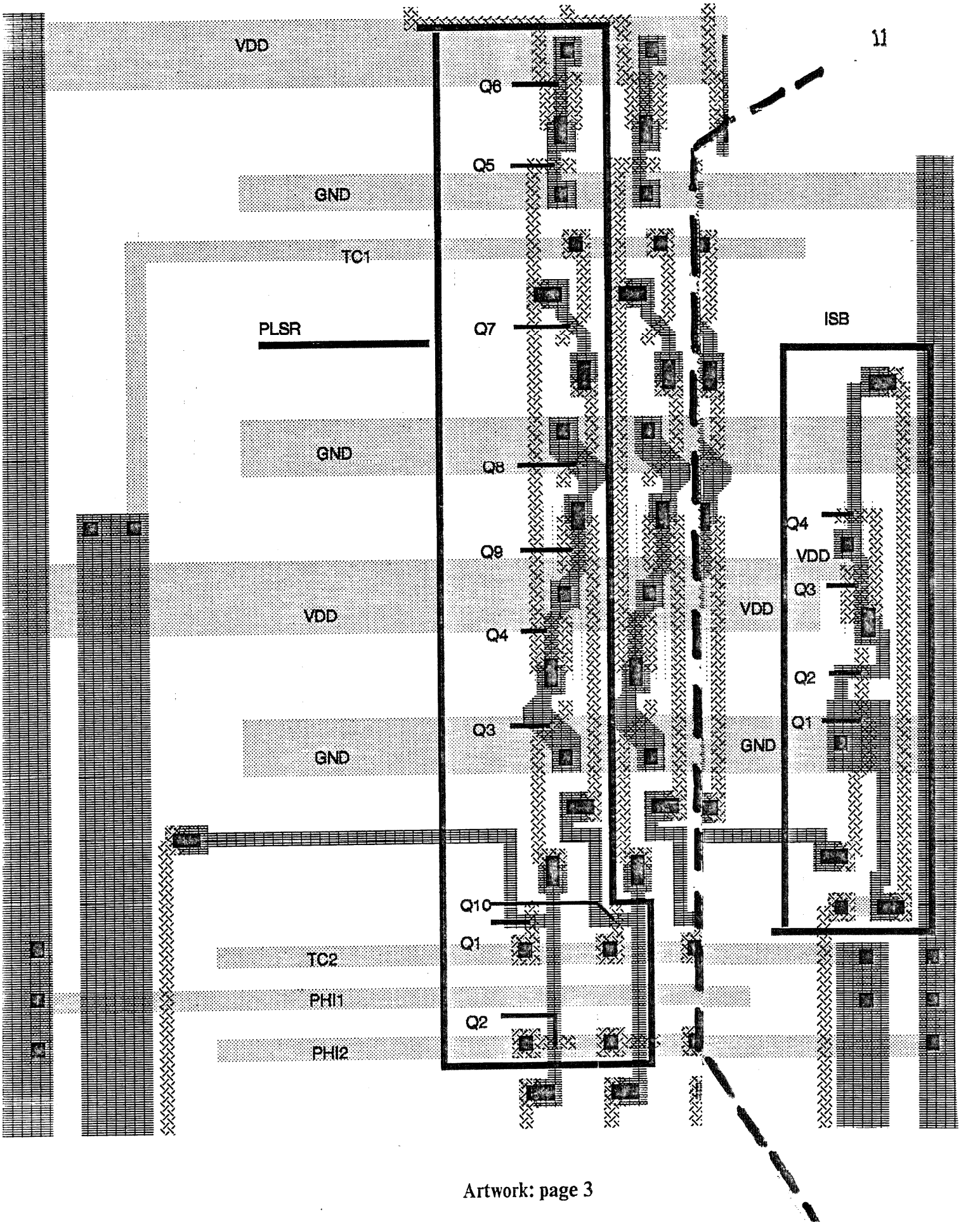


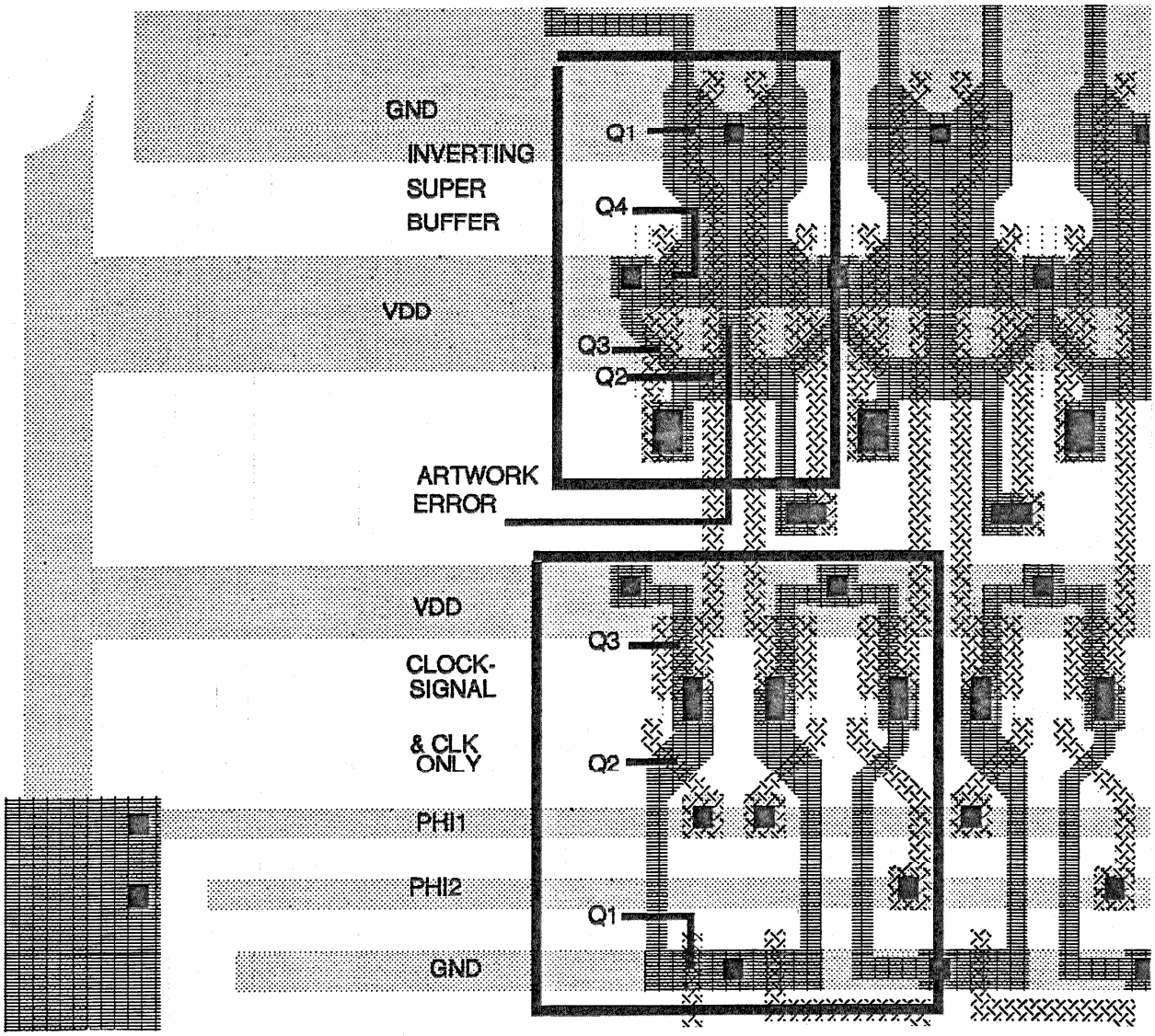
Entire Chip
Artwork: page 1

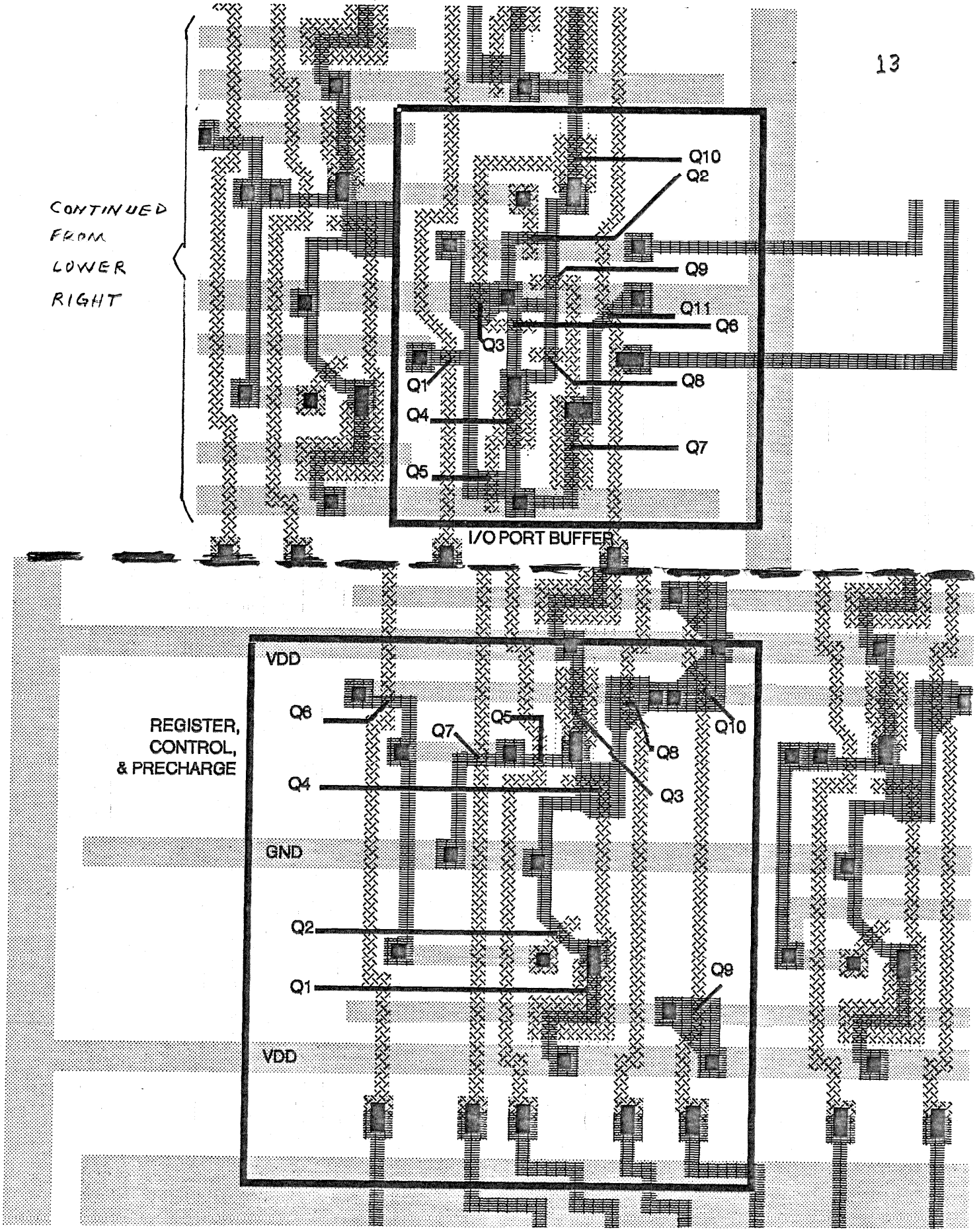
File: REG2BTT-1.prese | Scale: 97.00um/cm | Date: 7-Apr-82 10:32:16 | Page 1 of 1 [X: 1, Y: 1]



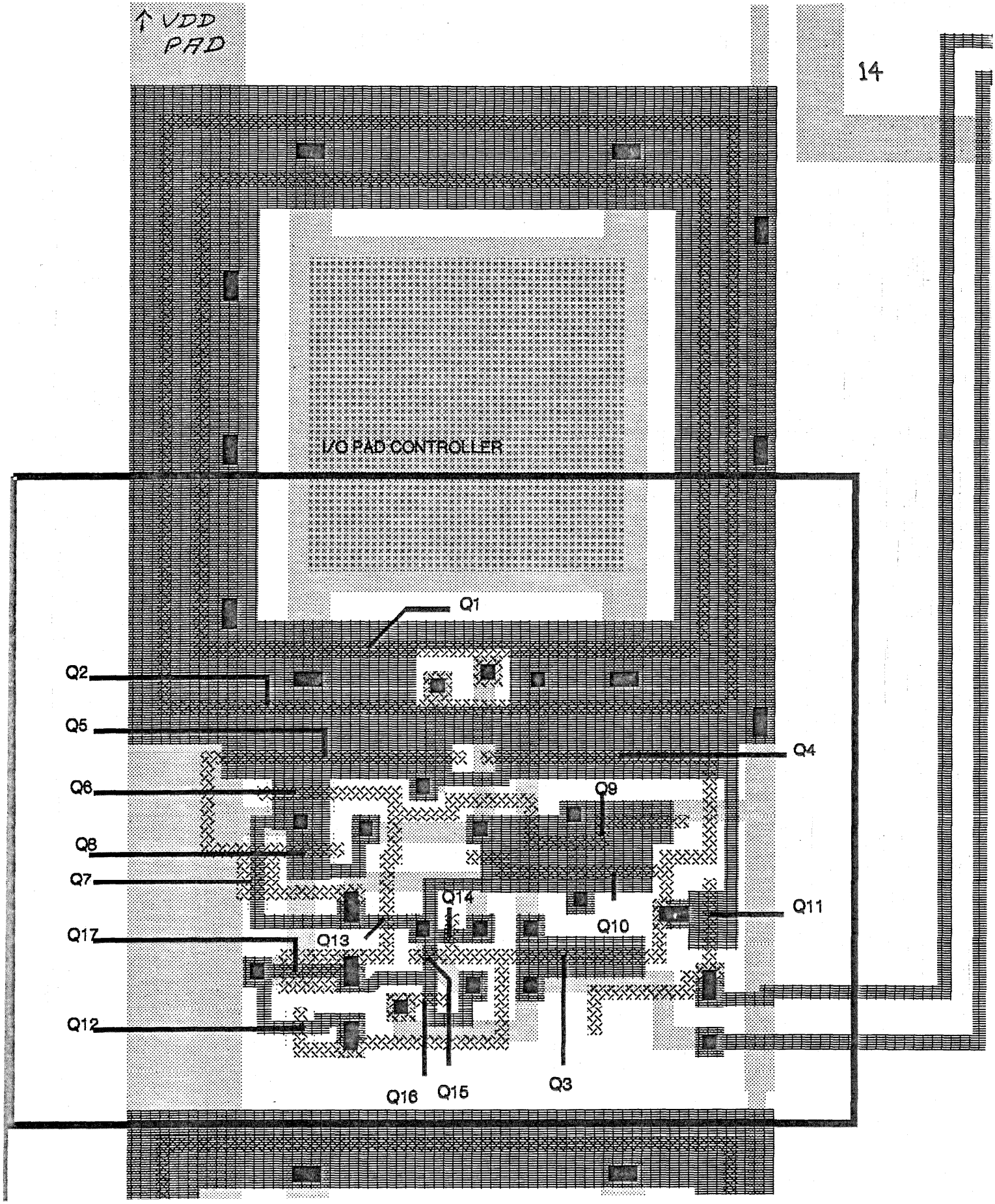
Artwork: page 2

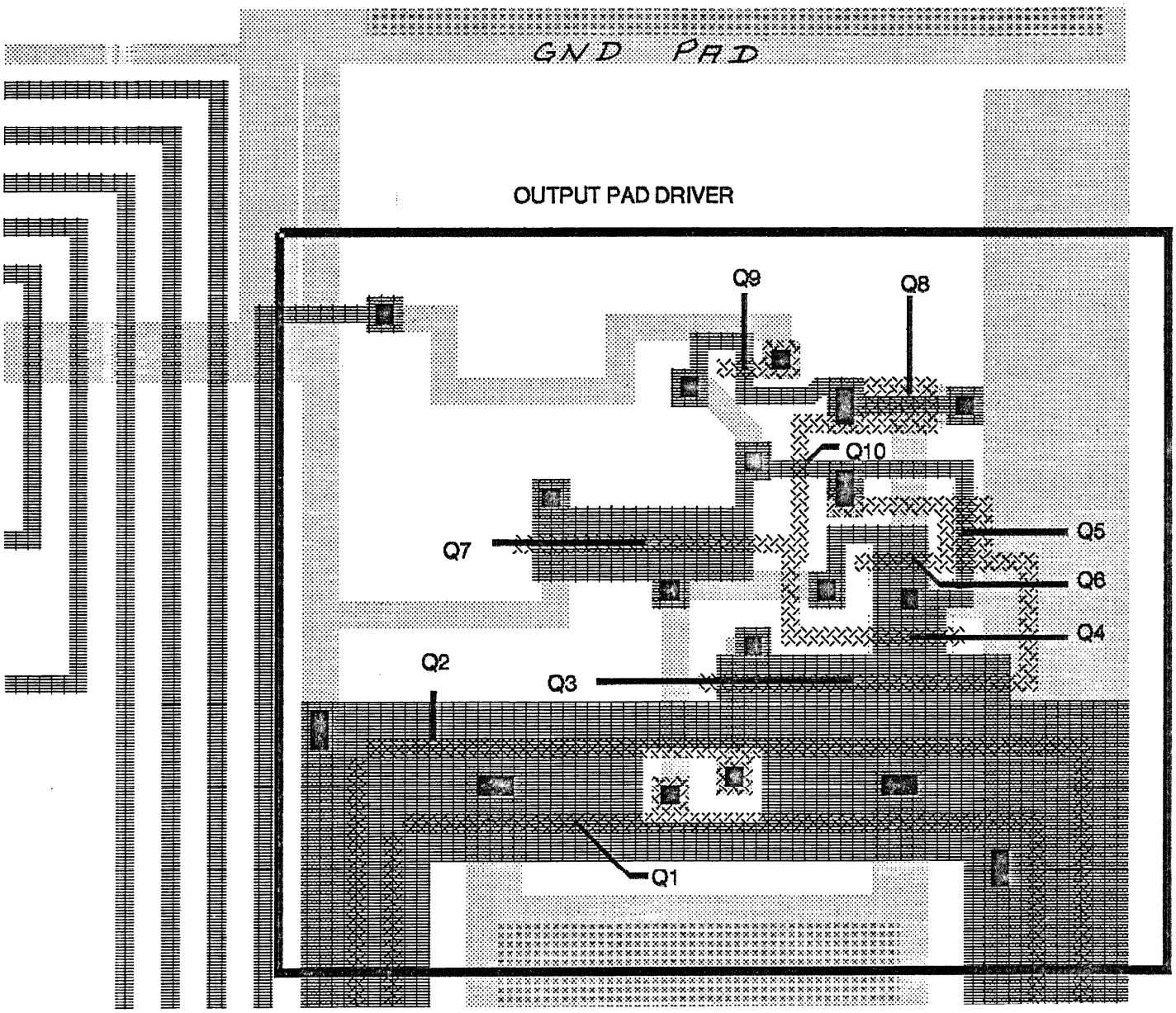

11
VDD
Q6
Q5
GND
TC1
PLSR
Q7
ISB
GND
Q8
Q4
Q9
VDD
VDD
Q3
Q4
Q2
Q3
Q1
GND
GND
Q10
Q1
TC2
PHI1
Q2
PHI2

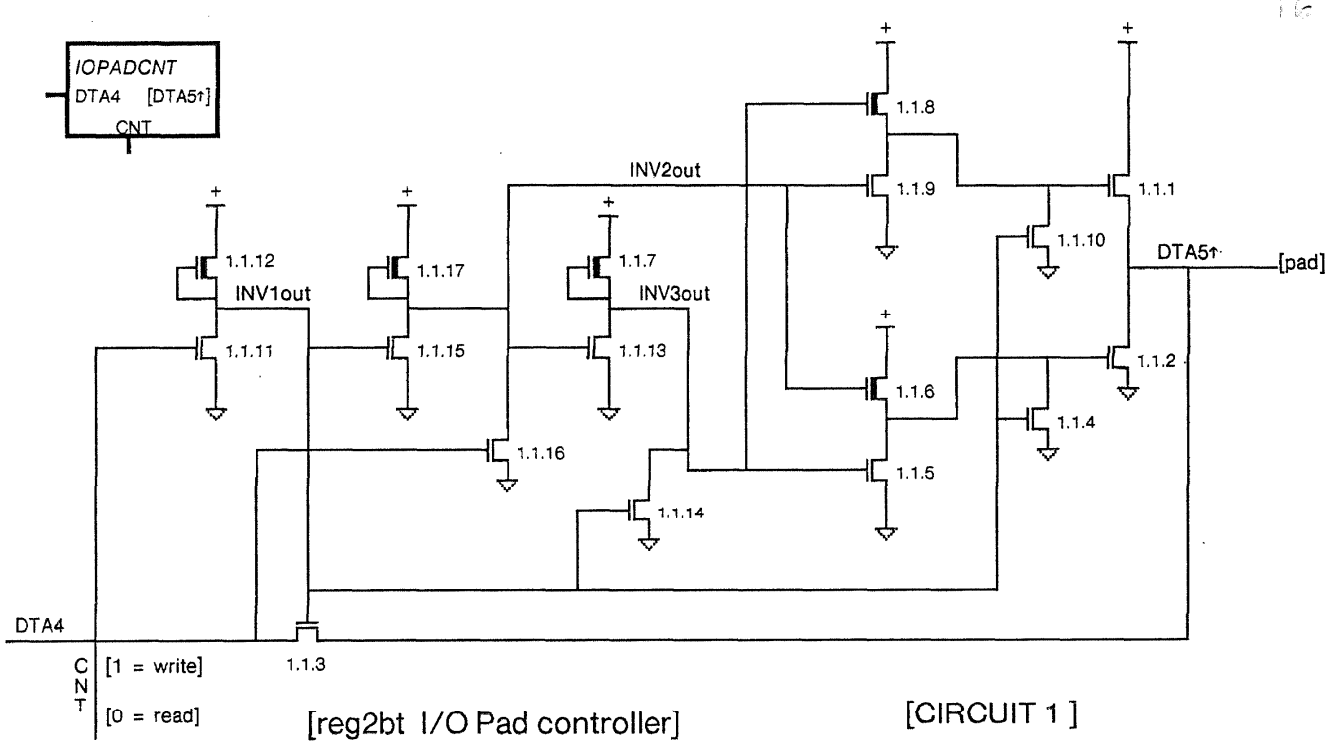

GND
INVERTING
SUPER
BUFFER
VDD
ARTWORK
ERROR
VDD
CLOCK-
SIGNAL
& CLK
ONLY
PHI1
PHI2
GND
Q1
Q4
Q3
Q2
Q3
Q2
Q1



Artwork: page 5

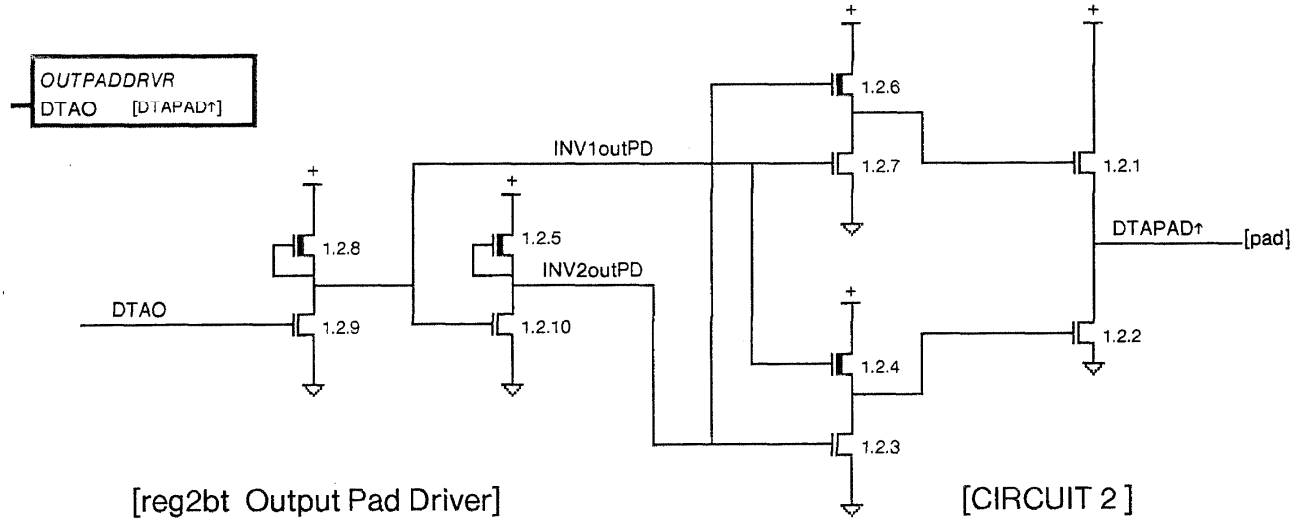

↑ VDD
PAD
14
I/O PAD CONTROLLER
Q1
Q2
Q5
Q4
Q6
Q9
Q8
Q7
Q14
Q11
Q17
Q13
Q10
Q12
Q16
Q15
Q3


GND PAD
OUTPUT PAD DRIVER
Q9
Q8
Q10
Q7
Q5
Q6
Q4
Q2
Q3
Q1



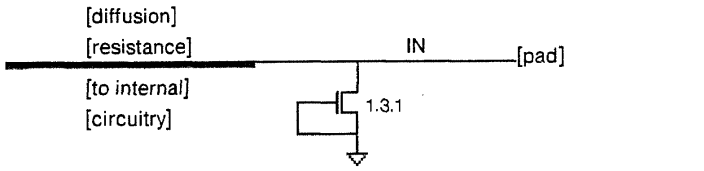
[reg2bt I/O Pad controller]

[CIRCUIT 1]



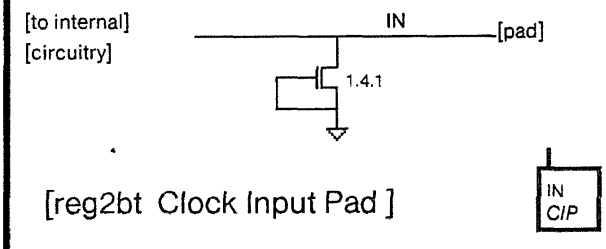
[reg2bt Output Pad Driver]

[CIRCUIT 2]



[reg2bt Signal Input Pad]

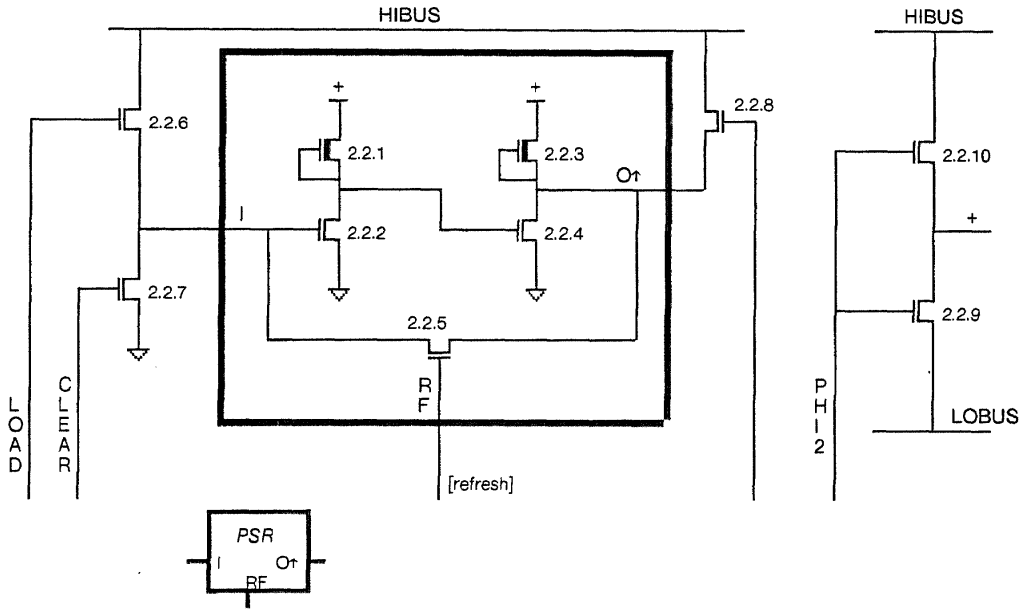
[CIRCUIT 3]



[reg2bt Clock Input Pad]

[CIRCUIT 4]

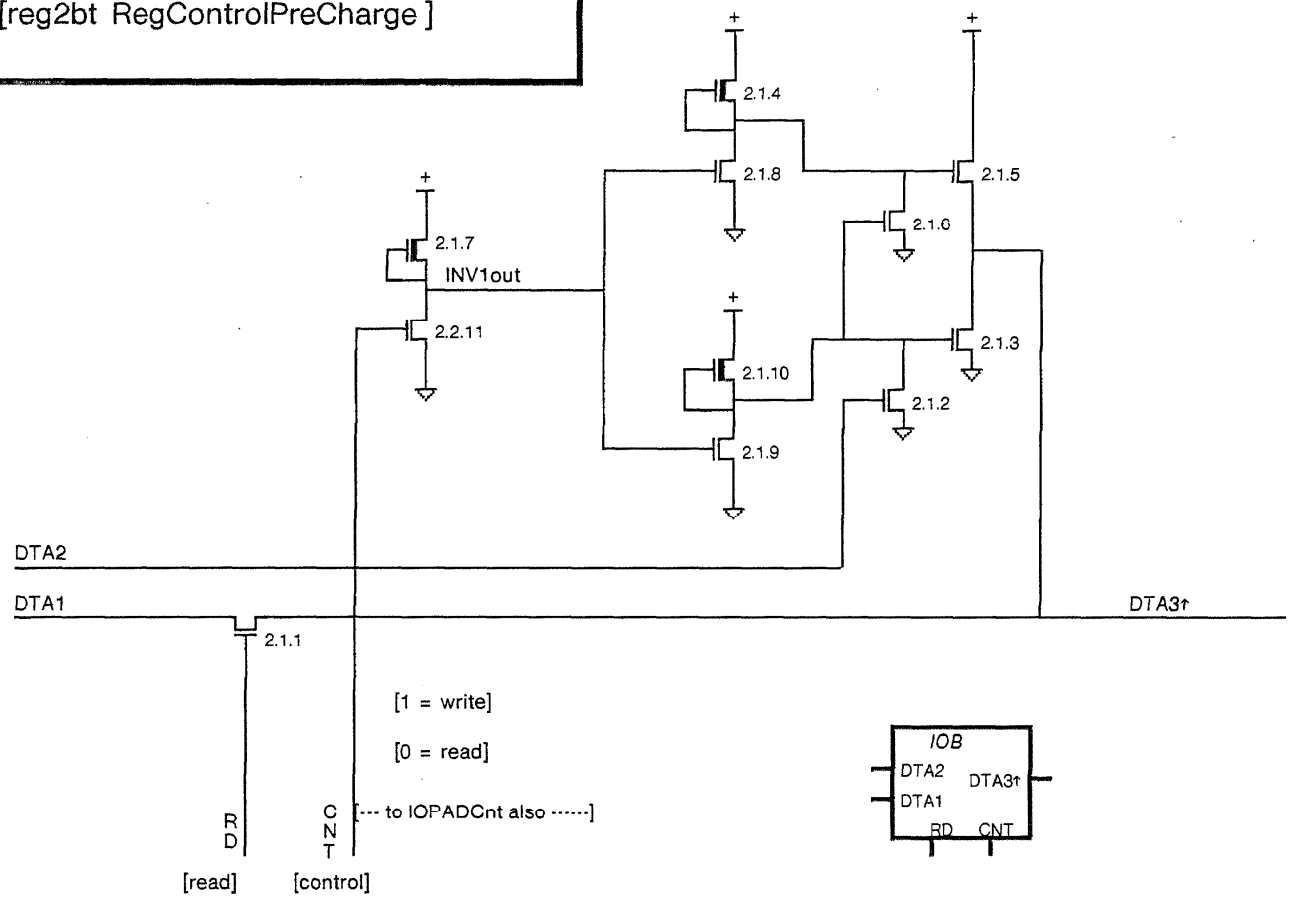
[reg01.sil] [McNair] [04/12/82]



[Pseudo Static Register]

[CIRCUIT 2]

[reg2bt RegControlPreCharge]

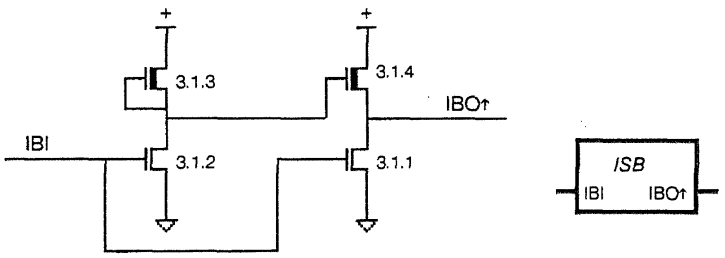


[read] [control]

[reg2bt I/O Port Buffer]

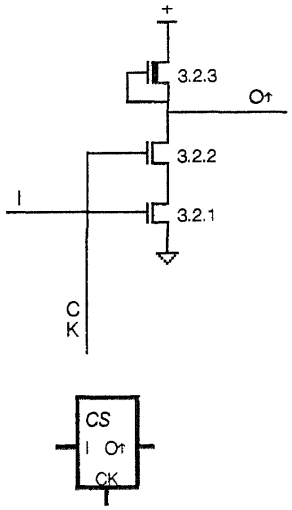
[CIRCUIT 1]

[reg02.sil] [McNair] [04/12/82]



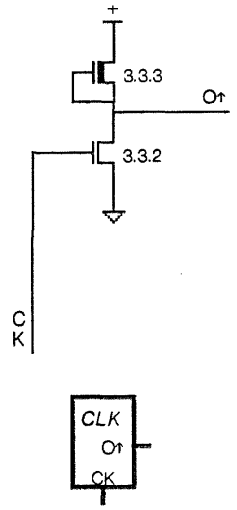
[reg2bt Inverting Super Buffer]

[CIRCUIT 1]



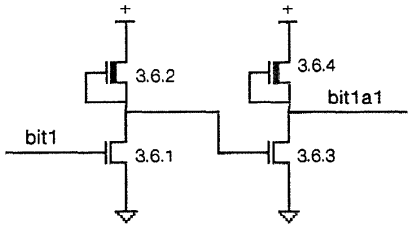
[reg2bt Clock-Signal]

[CIRCUIT 2]



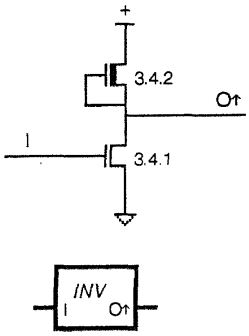
[reg2bt Clk Only]

[CIRCUIT 3]



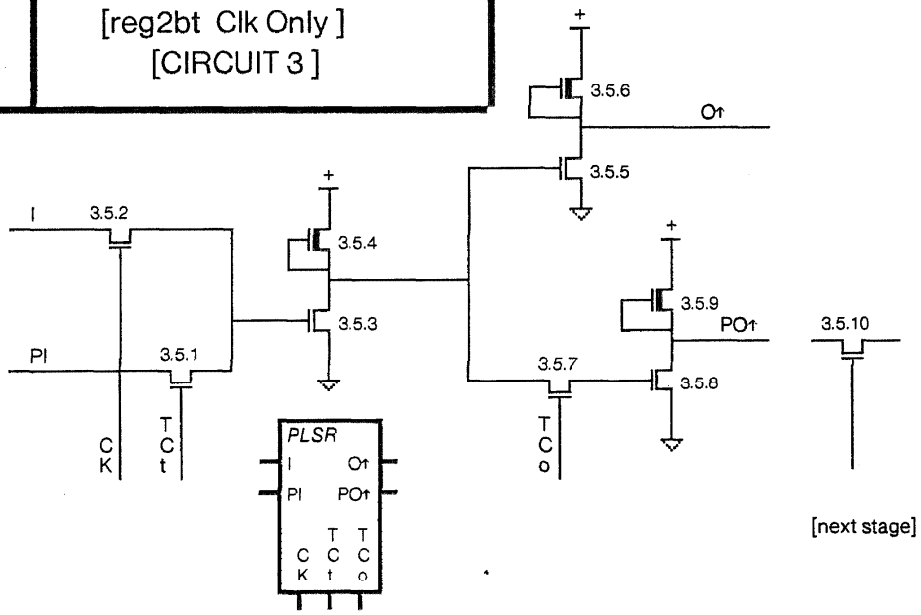
[Decode Example]

[CIRCUIT 6]



[reg2bt Inverter]

[CIRCUIT 4]



[reg2bt PLSR parallel shift reg]

[CIRCUIT 5]

[reg03.sil] [McNair] [04/21/82]

Composite Schematic

The composite schematic is now presented using the simplified logic symbols shown earlier. The next page, REG2bt Macro Over View, shows the logic symbols, called macros, that are to be used in the composite schematic. The figure is organized to show the basic floor plan of the chip and to reference to the artwork and schematic detail pages. The composite schematic itself is shown as two pages, RegTot.01.sil and RegTot.02.sil.

The composite schematic uses signal names with slight variations from ones shown elsewhere. The names are equivalent if they are similar,
e.g. PLSRIN = PLSR-IN, PHI-1 = PHIone = [PHI1] .

The basic control signals are assigned names as they move through the logic. There is some interest in being able to compare the top of RegTot.02.sil directly to the design statement. Therefore, a brief explanation of these signal names is provided. The signals start at the bottom of page RegTot.01.sil as:

BIT1 [CLEAR]
BIT2 [LOAD]
BIT3 [OP] i.e. OutPut
BIT4 [IOC-IN]
BIT5 [IOC-BUS-RD]

These signal may generate more than one control line going to datapath and signals may also come out of the decoder inverted. To handle these issues as well as the logic level, the following signal naming conventions are used:

BTnLEVELoccurance(inversion)(clocking)

where n is the bit number as shown above

LEVEL is Λ out of the decoder, B out of the PLSR, C out of the clocks and D out of the Inverting Super Buffers (and ready for the datapath)

inversion, if present, is specified by B (mnemonic for Bar)

clocking, if present, is &# where # refers to clock PHI-# and the order of &# show the order of PHI-1 and PHI-2 clocking

e.g.
BT2D2B&1&2 is bit2 (load), it is at the *ISB* output logic level (D), it is the second use of bit2, it is inverted (B), i.e. "NOT load" and it was clocked first by PHI-1 and then PHI-2.

The second level of RegTot.01.sil shows the decoder. The top 8 inverters ***start the schematic convention of being physically positioned similar to the artwork.***

Let us consider the operation of the datapath seen at the top of RegTot.02.sil. It can be seen (see next paragraph) that the data input from the pad goes to the register the BB design statement called OUTPUT-REGISTER (datapath elements are related to BB design statements names at the very top of the schematic). This was a major surprise.

Tracing the "data input" may be done as follows. By noting connection point names [DTA5↑], DTA4, DTA3↑ and DTA1 and looking at detailed schematics for IOPADCNT and IOB, one can see that pass transistors 1.1.3 and 2.1.1 route the pad signal to the input (I) of the "OUTPUT-REGISTER"

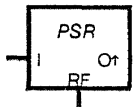
The use of this composite schematic to aid in understanding is continued in the next section in which the MOSSIM II simulation is described.

DataPath

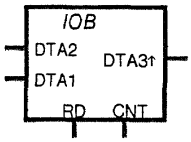
Schematics: REG02.sil

Artwork: page 5

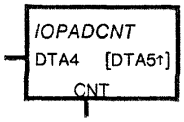
Pseudo Static Register



I/O Buffer



I/O
Pad
Controller



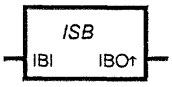
Artwork: page 6

BUFFERS and Clocking

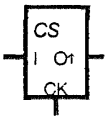
Schematics: REG03.sil

Artwork: page 4

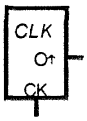
Inverting Super Buffer



Clock&Signal



Clock (only)



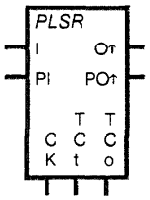
Total Chip part 2 RegTot.02.sil

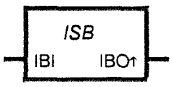
PLSR

Schematics: REG03.sil

Artwork: page 3

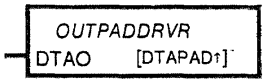
Parallel Load Shift Register





Inverting Super Buffer

Output
Pad
Driver



Artwork: page 7

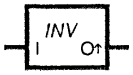
Total Chip part 1 RegTot.01.sil

DECODE

Schematics: REG03.sil

Artwork: page 2

Inverter



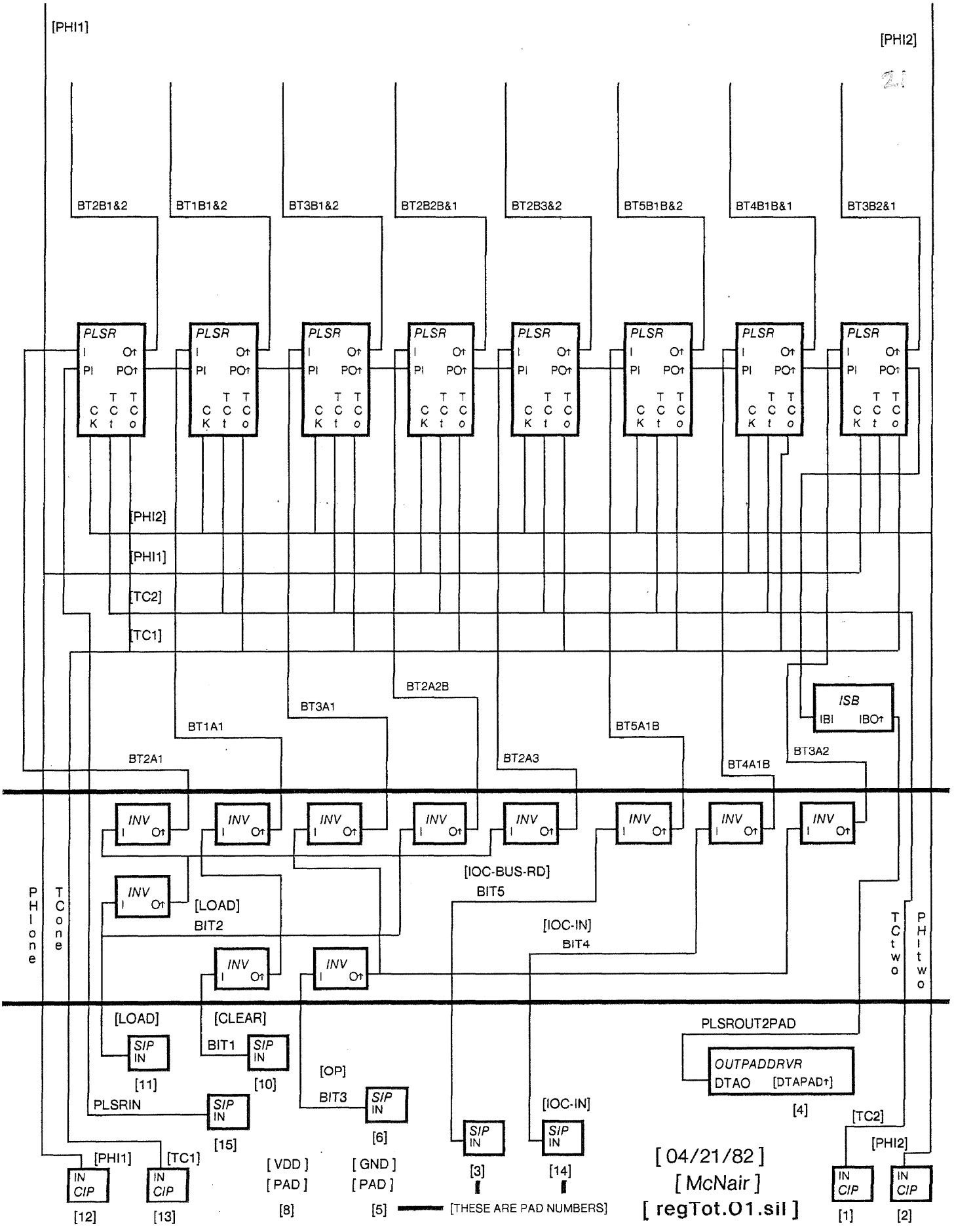
Signal Input Pad

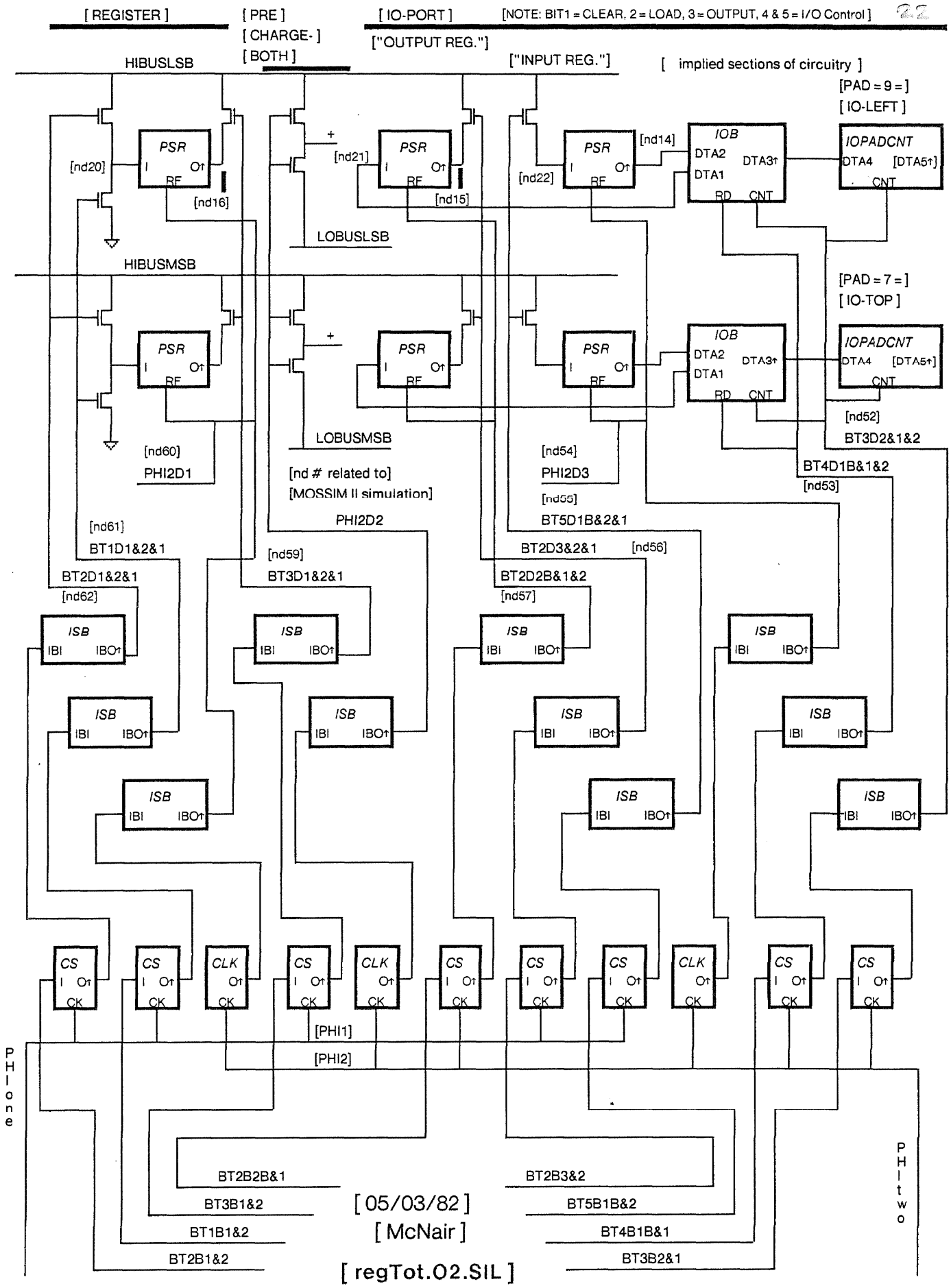
Clock Input Pad

PADS

Schematics: REG01.sil

REG2bt Macro Over View


[PHI1]
[PHI2]
21
BT2B1&2
BT1B1&2
BT3B1&2
BT2B2B&1
BT2B3&2
BT5B1B&2
BT4B1B&1
BT3B2&1
PLSR
I
O↑
PI
PO↑
CK
Tt
TCo
[PHI2]
[PHI1]
[TC2]
[TC1]
ISB
IBI
IBO↑
BT2A2B
BT3A1
BT1A1
BT5A1B
BT2A1
BT2A3
BT4A1B
BT3A2
INV
[IOC-BUS-RD]
BIT5
[LOAD]
BIT2
[IOC-IN]
BIT4
PHIone
TCone
TCtwo
PHItwo
[LOAD]
[CLEAR]
BIT1
SIP
IN
[11]
[10]
[OP]
BIT3
[6]
PLSROUT2PAD
OUTPADDRVR
DTAO
[DTAPAD↑]
[4]
PLSRIN
[15]
[IOC-IN]
[3]
[14]
[TC2]
[PHI2]
[PHI1]
[TC1]
IN
CIP
[12]
[13]
[VDD]
[PAD]
[8]
[GND]
[PAD]
[5]
[THESE ARE PAD NUMBERS]
[04/21/82]
[McNair]
[regTot.01.sil]
[1]
[2]


[REGISTER]
[PRE]
[CHARGE-]
[BOTH]
[IO-PORT]
["OUTPUT REG."]
["INPUT REG."]
[NOTE: BIT1 = CLEAR, 2 = LOAD, 3 = OUTPUT, 4 & 5 = I/O Control]
22
[implied sections of circuitry]
HIBUSLSB
HIBUSMSB
LOBUSLSB
LOBUSMSB
[PAD = 9 =]
[IO-LEFT]
[PAD = 7 =]
[IO-TOP]
PSR
IOB
IOPADCNT
ISB
CS
CLK
[nd20]
[nd21]
[nd22]
[nd14]
[nd15]
[nd16]
[nd52]
[nd53]
[nd54]
[nd55]
[nd56]
[nd57]
[nd59]
[nd60]
[nd61]
[nd62]
PHI2D1
PHI2D2
PHI2D3
BT1D1&2&1
BT2D1&2&1
BT3D1&2&1
BT2D3&2&1
BT2D2B&1&2
BT5D1B&2&1
BT4D1B&1&2
BT3D2&1&2
[nd # related to]
[MOSSIM II simulation]
[PHI1]
[PHI2]
PHIone
PHItwo
BT2B2B&1
BT3B1&2
BT1B1&2
BT2B1&2
BT2B3&2
BT5B1B&2
BT4B1B&1
BT3B2&1
[05/03/82]
[McNair]
[regTot.02.SIL]

Brief Functional Description and MOSSIM II Simulation of REG2bt

The next page presents a timing diagram for REG2bt. The diagram will be used to explain a little about the chip operation and to help convey some information about the simulation.

The MOSSIM II logic simulation (Ref. MSII) was fairly easy to set up. Automatic extraction of the circuit from the CIF file was used (via the EXTRACT program - Ref. EXT). EXTRACT did most of the work of creating the 292 transistor network. No equivalent circuits were needed. A few simple lines of instruction to the CONVERT program made the conversion from the EXTRACT format to the MOSSIM II network format. The actual details of these operations are presented later to provide a fully documented example. Many readers may elect to not look at the detail.

The simulation "ran" the chip through an entire cycle of clearing the register, taking in data, resetting the IO pads to insure that the output was truly coming from "stored" data, and then outputting the data. The simulation elected to "run" the chip without using signals that split any clock phase. Using split phase operation might allow data movement in fewer clock segments.

The simulation drove both a "zero" and "one" to test both the precharge ability to provide a "one" and the pull down circuitry's ability to override the precharge.

The timing diagram shows the clock phases as though they were not required to be non-overlapping. This was just a convenience. The chip and the simulation actually use non-overlapping clocks.

It *is to be accepted* that the timing diagram represents the simulation results. Those electing to have a careful look at the simulation may want to look back and confirm that the diagram represents the simulation printout.

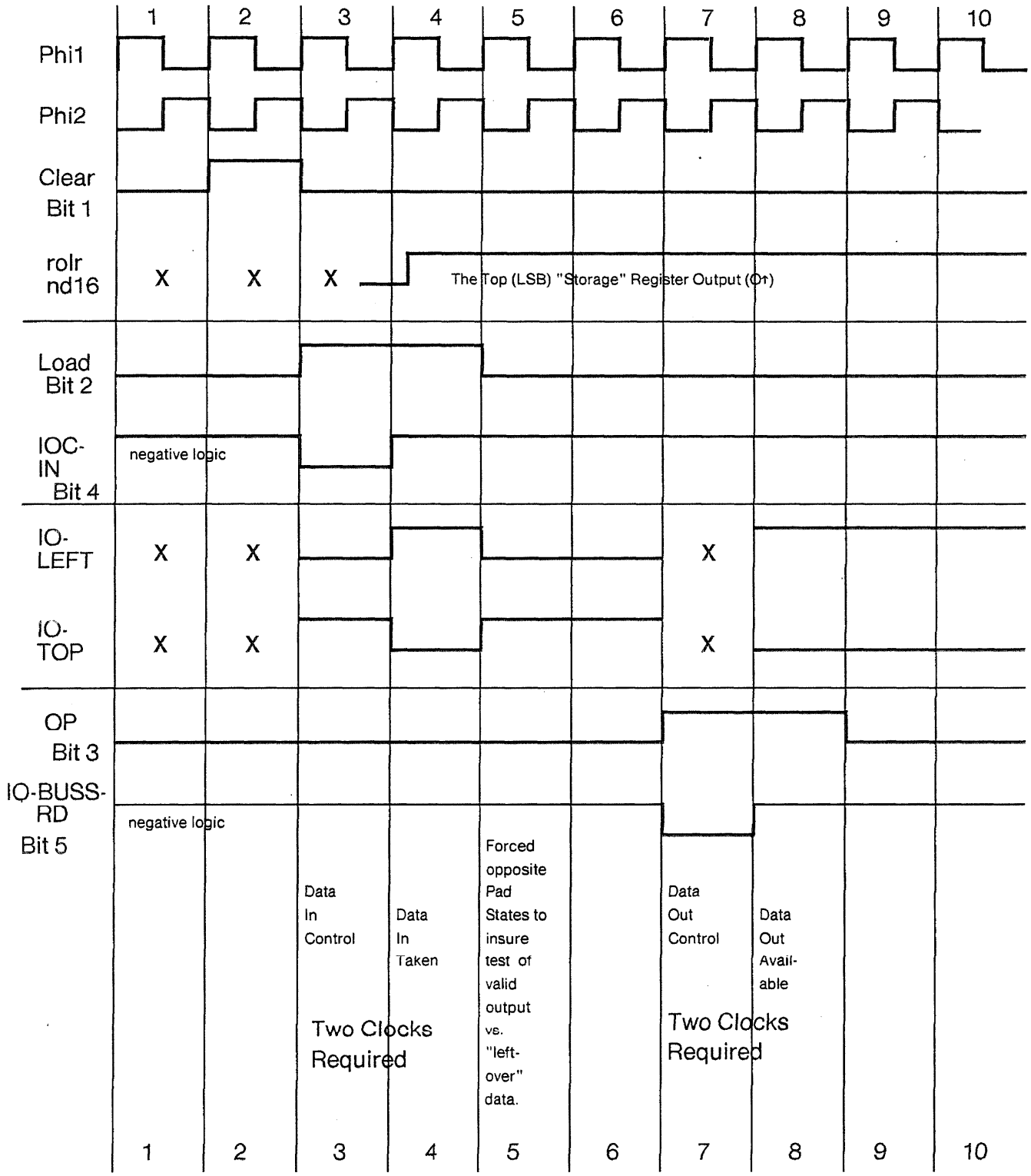
In this simulation, most of the signals shown in the timing diagram were (in effect) applied to the center metal of the pad and had to propagate their effect through the entire chip from input pads back to output pads.

One cycle of simulation was performed just to see the unknown states that appear initially in much of the chip. One signal, `rolr,nd16` is shown in the timing diagram as a typical example to use in the discussion of the CLEAR operation. This signal is actually the first bit of a MOSSIM II vector printout and is identified more rigorously in later discussion. For now, it suffices to say that the signal corresponds to the output (O↑) of the upper left "register" in figure RegTot.02.sil.

The CLEAR signal was set true during clock 2. It was expected that `rolr,nd16` would be set to zero by the end of clock 2. That was wrong. The simulation renewed the realization that there was a clocking delay in getting the CLEAR signal from the PAD to the data path. There was a one clock delay between the CLEAR being applied and it being effective.

The simulation also showed that there was a one clock delay between the control signal setting up data input and the actual taking of the data. This is shown by the load, IOC-IN, IO-LEFT and IO-TOP signals in the timing diagram section showing the actions at clocks three and four.

The clocking scheme also required two clocks to get data out, as seen in the timing diagram section showing the actions at clocks seven and eight.



Reg2btTiming.sil

03/30/82

The MOSSIM II simulation effort consisted of the following steps:

* Correct Artwork in one area (see prior "Artwork: page 4" discussion). Corrections reflected by use of filename REG2btFix
* Find PAD positions - proportional calculations relating BB units to the scaling of a composite plot is recommended.
* Add Signal naming (of the format shown below) to the BB CIF file.
* Run the EXTRACT program - example shown below.
* Run the CONVERT program - example shown below.
* Make a MOSSIM II run control file (or run interactively).
* Run the MOSSIM II program - example shown below.

The concept of proportional calculations relating BB units to the scaling of a composite plot are explained in Appendix B relative to determining the x,y coordinates shown for PHI-1 in the signal names added to the CIF file (shown below). These names are added to the very last DS segment of the CIF file after the call that translates the entire chip for 0,0 as an origin coordinate. The syntax for the names is as described in Reference EXT.

```
LNM;
84 100 1560 VDD ;
84 833 1560 GND ;
84 100 1233 IO-LEFT A;
84 100 967 CLEAR IA;
84 100 667 LOAD IA;
84 145 390 PHI-1 IA;
84 150 250 TC-1 IA;
84 350 100 IOC-IN IA;
84 567 100 PLSR-IN IA;
84 833 600 IOC-BUSS-RD IA;
84 800 383 PHI-2 IA;
84 800 250 TC-2 IA;
84 833 1167 PLSR-OUT A;
84 567 1567 OP IA;
84 350 1567 IO-TOP A;
```

The running of the EXTRACT program (Ref. EXT) for Reg2bt is shown below. Most of the typing shown below was provided by the computer. The user typed only the lines starting with @ or *. The running of the program "extracted" from the CIF layout file the information necessary to identify every transistor, its type and connection points (nodes). The program also determined all the nodes in the interconnection and the area parameters for the nodes and transistors.

```
[Recording initiated at Thu 11-Mar-82 15:23:13]

LINK FROM DICKM, TTY 6

 TOPS-20 Command processor 5(625)
@EXTRACT
EXTRACT Rev 1.7   Feb 4, 1982

*cif reg2btfix
Last Cell Defined: 153
0 errors, 0 warnings, 2304 commands
283 single calls, 224 additional calls
*set diag
*set location
*set time
.00 Seconds CPU, .00 Seconds Elapsed
*extract 153
Enter Extraction Output File: reg2btfixe.xtr
Enter Diagnostic Output File: reg2btfixe.diagnostic
Total Scan Lines: 841 Width: 468
Extracting 1 bands of 841 lines:
1
160 Nodes, 292 Total Transistors
79 Null(deleted), 0 One Contact, 292 Two Contact, 0 Three(or more) Contact
102 Depletion, 190 Enhancement, 9 Implant Errors

84 Pullups, 131 Pulldowns, 50 Pass Transistors,
18 Depletion to VDD, 0 Yellow Transistors, 9 Prechargers
64.39 Seconds CPU, 149.64 Seconds Elapsed
*EXIT

@type reg2btfixe.diagnostic

Extracting Cell 153 from File reg2btfix.cif at Scale 300

160 Nodes, 292 Total Transistors
79 Null(deleted), 0 One Contact, 292 Two Contact, 0 Three(or more) Contact
102 Depletion, 190 Enhancement, 9 Implant Errors
Node Always High -NODE27 @ 653.99 1845.01
Node Always High -NODE43 @ 656.99 1674.01
Unused Node PLSR-OUT @ 1280.99 1869.01

84 Pullups, 131 Pulldowns, 50 Pass Transistors,
18 Depletion to VDD, 0 Yellow Transistors, 9 Prechargers
```

The running of the CONVERT program (written at CIT by Mike Schuster) for Reg2bt is shown below. Most of the typing shown below was provided by the computer. The user typed only the lines starting with @ or >. The running of the program "converted" from the EXTRACT format to the network format used by the MOSSIM II program. Operations "Strength" and "size" were done to set up network characteristics as described in the MOSSIM II User's Manual (Ref. MSII).

```
[Recording initiated at Thu 11-Mar-82 16:10:15]

LINK FROM DICKM, TTY 6

 TOPS-20 Command processor 5(625)
@convert.EXE.3
Convert, Version 1
>read reg2btfixe
  160 nodes, 292 transistors
>strength *
  2 strengths assigned
>size share:2
precedence error: -node126=k1(319) -node119=k1(605) -node142=k1(843)
precedence error: -node127=k1(319) -node120=k1(605) -node141=k1(781)
precedence error: -node128=k1(319) -node123=k1(605) -node139=k1(700)
precedence error: -node131=k1(349) -node124=k1(605) -node143=k1(799)
  3 sizes assigned
>write reg2btfixe
>q
```

The running of the MOSSIM II program (Ref. MSII) for Reg2bt is shown below. Much of the typing shown below was provided by the computer. The user typed only the lines starting with @ or >. Almost all of the lines starting with > were taken from a control file that actually directed the MOSSIM II run. Therefore, run iterations (and there were some) required only simple edits to the control file and then a new run could be made.

The next paragraph shows the relation of some of the nodes in the layout to the Composite Schematic. The layout nodes and some transistors were identified by using "tracing" features of the CONVERT program. These features proved to be very helpful in getting enough reference points in the artwork to allow real "debugging" of the simulation. The "tracing" features of the CONVERT program are exemplified in an example following the simulation.

MOSSIM II NODE NAMES vs. REGToT.02.sil Names

```
vector    nodes by    Schematic Relation (nodes also shown on Sch.)
name      Extract's #'s

rilr   nd 20 21  22    no names, connection I of PSR of LSB (Top)
rolr   nd 16 15  14    no names, connection O↑ of PSR of LSB (Top)
rflr   nd 60 57  54    PHI2D1, BT2D2B&1&2, PHI2D3
rdlr   nd 62 55        BT2D1&2&1, BT5D1B&2&1
wrlr   nd 59 56        BT3D1&2&1, BT2D3&2&1
cntlr  nd 61 53  52    BT1D1&2&1, BT4D1B&1&2, BT3D2&1&2
```

The trailing `lr` in the above names simply means left to right in the artwork. The simulation seen below also reflects two actions taken to simplify typing:
a) the file Reg2btFixe.ntwk was copied to the simple name re.ntwk and
b) in that file, the node names of the form `-node#` were abbreviated to `nd#`.

```
[Recording initiated at Tue 6-Apr-82 15:18:36]

LINK FROM DICKM, TTY 10

 TOPS-20 Command processor 5(625)
@mossim
Mossim II, Version 1
>so rs
>comment this file (rs.source) for working with reg2btfixe.ntwk
>sw statistics:1
>
>read re.ntwk
    REG2BTFIX.cif #153 NMOS
    160 nodes, 292 transistors, 0 blocks
>clock phi-1:0100 phi-2:0001
>
>vector rilr nd20 nd21 nd22
>vector rolr nd16 nd15 nd14
>vector rflr nd60 nd57 nd54
>vector rdlr nd62 nd55
>vector wrlr nd59 nd56
>vector cntlr nd61 nd53 nd52
>
>watch /4 IO-LEFT
>watch /4 IO-TOP
>watch /4 CLEAR
```

```
>watch /4 LOAD
>watch /4 PHI-1
>watch /4 TC-1
>watch /4 IOC-IN
>watch /4 PLSR-IN
>watch /4 IOC-BUSS-RD
>watch /4 PHI-2
>watch /4 TC-2
>watch /4 PLSR-OUT
>watch /4 OP
>watch /4 nd13 nd29
>
>watch /* rilr rolr rflr rdlr wrlr cntlr
>
>set IO-LEFT:0
>set IO-TOP:0
>set CLEAR:0
>set LOAD:0
>set TC-1:0
>set IOC-IN:1
>set PLSR-IN:0
>set IOC-BUSS-RD:1
>set TC-2:0
>set PLSR-OUT:0
>set OP:0
>cycle
  1.1| rilr:XXX rolr:XXX rflr:000 rdlr:00 wrlr:00 cntlr:000
  1.2| rilr:XXX rolr:XXX rflr:000 rdlr:XX wrlr:XX cntlr:X00
  1.3| rilr:XXX rolr:XXX rflr:000 rdlr:00 wrlr:00 cntlr:000
  1.4| IO-LEFT:X IO-TOP:X CLEAR:0 LOAD:0 PHI-1:0 TC-1:0 IOC-IN:1 PLSR-IN:0
     | IOC-BUSS-RD:1 PHI-2:1 TC-2:0 PLSR-OUT:X OP:0 nd13:1 nd29:1 rilr:XXX
     | rolr:XXX rflr:111 rdlr:00 wrlr:00 cntlr:000
    1.25 seconds, 16 steps, 319 events, 136 state changes
>set CLEAR:1
>cy
  2.1| rilr:XXX rolr:XXX rflr:000 rdlr:00 wrlr:00 cntlr:000
  2.2| rilr:XXX rolr:XXX rflr:000 rdlr:00 wrlr:00 cntlr:000
  2.3| rilr:XXX rolr:XXX rflr:000 rdlr:00 wrlr:00 cntlr:000
  2.4| IO-LEFT:X IO-TOP:X CLEAR:1 LOAD:0 PHI-1:0 TC-1:0 IOC-IN:1 PLSR-IN:0
     | IOC-BUSS-RD:1 PHI-2:1 TC-2:0 PLSR-OUT:X OP:0 nd13:1 nd29:1 rilr:XXX
     | rolr:XXX rflr:111 rdlr:00 wrlr:00 cntlr:000
    .40 seconds, 9 steps, 82 events, 35 state changes
>set CLEAR:0
>force IO-LEFT:0
>force IO-TOP:1
>set IOC-IN:0
>set LOAD:1
>cy
  3.1| rilr:XXX rolr:XXX rflr:000 rdlr:00 wrlr:00 cntlr:000
  3.2| rilr:0XX rolr:0XX rflr:000 rdlr:00 wrlr:00 cntlr:100
  3.3| rilr:0XX rolr:0XX rflr:000 rdlr:00 wrlr:00 cntlr:000
  3.4| IO-LEFT:0 IO-TOP:1 CLEAR:0 LOAD:1 PHI-1:0 TC-1:0 IOC-IN:0 PLSR-IN:0
     | IOC-BUSS-RD:1 PHI-2:1 TC-2:0 PLSR-OUT:X OP:0 nd13:1 nd29:1 rilr:00X
     | rolr:00X rflr:101 rdlr:00 wrlr:00 cntlr:010
    .84 seconds, 16 steps, 96 events, 70 state changes
>force IO-LEFT:1
>force IO-TOP:0
>set IOC-IN:1
>cy
```

```
  4.1| rilr:01X rolr:01X rflr:000 rdlr:00 wrlr:00 cntlr:000
  4.2| rilr:11X rolr:11X rflr:000 rdlr:10 wrlr:01 cntlr:000
  4.3| rilr:11X rolr:11X rflr:000 rdlr:00 wrlr:00 cntlr:000
  4.4| IO-LEFT:1 IO-TOP:0 CLEAR:0 LOAD:1 PHI-1:0 TC-1:0 IOC-IN:1 PLSR-IN:0
     | IOC-BUSS-RD:1 PHI-2:1 TC-2:0 PLSR-OUT:X OP:0 nd13:1 nd29:1 rilr:11X
     | rolr:11X rflr:101 rdlr:00 wrlr:00 cntlr:000
    .43 seconds, 14 steps, 83 events, 52 state changes
>force IO-LEFT:0
>force IO-TOP:1
>set LOAD:0
>cy
  5.1| rilr:11X rolr:11X rflr:000 rdlr:00 wrlr:00 cntlr:000
  5.2| rilr:11X rolr:11X rflr:000 rdlr:10 wrlr:01 cntlr:000
  5.3| rilr:11X rolr:11X rflr:000 rdlr:00 wrlr:00 cntlr:000
  5.4| IO-LEFT:0 IO-TOP:1 CLEAR:0 LOAD:0 PHI-1:0 TC-1:0 IOC-IN:1 PLSR-IN:0
     | IOC-BUSS-RD:1 PHI-2:1 TC-2:0 PLSR-OUT:X OP:0 nd13:1 nd29:1 rilr:11X
     | rolr:11X rflr:111 rdlr:00 wrlr:00 cntlr:000
    .43 seconds, 14 steps, 70 events, 51 state changes
>unforce IO-LEFT
>unforce IO-TOP
>cy
  6.1| rilr:11X rolr:11X rflr:000 rdlr:00 wrlr:00 cntlr:000
  6.2| rilr:11X rolr:11X rflr:000 rdlr:00 wrlr:00 cntlr:000
  6.3| rilr:11X rolr:11X rflr:000 rdlr:00 wrlr:00 cntlr:000
  6.4| IO-LEFT:0 IO-TOP:1 CLEAR:0 LOAD:0 PHI-1:0 TC-1:0 IOC-IN:1 PLSR-IN:0
     | IOC-BUSS-RD:1 PHI-2:1 TC-2:0 PLSR-OUT:X OP:0 nd13:1 nd29:1 rilr:11X
     | rolr:11X rflr:111 rdlr:00 wrlr:00 cntlr:000
    .36 seconds, 8 steps, 44 events, 26 state changes
>set OP:1
>set IOC-BUSS-RD:0
>cy
  7.1| rilr:11X rolr:11X rflr:000 rdlr:00 wrlr:00 cntlr:000
  7.2| rilr:11X rolr:11X rflr:000 rdlr:00 wrlr:00 cntlr:000
  7.3| rilr:11X rolr:11X rflr:000 rdlr:00 wrlr:00 cntlr:000
  7.4| IO-LEFT:X IO-TOP:X CLEAR:0 LOAD:0 PHI-1:0 TC-1:0 IOC-IN:1 PLSR-IN:0
     | IOC-BUSS-RD:0 PHI-2:1 TC-2:0 PLSR-OUT:X OP:1 nd13:1 nd29:1 rilr:11X
     | rolr:11X rflr:111 rdlr:00 wrlr:00 cntlr:001
    .39 seconds, 17 steps, 100 events, 81 state changes
>set OP:1
>set IOC-BUSS-RD:1
>cy
  8.1| rilr:11X rolr:11X rflr:000 rdlr:00 wrlr:00 cntlr:000
  8.2| rilr:111 rolr:111 rflr:000 rdlr:01 wrlr:10 cntlr:000
  8.3| rilr:111 rolr:111 rflr:000 rdlr:00 wrlr:00 cntlr:000
  8.4| IO-LEFT:1 IO-TOP:0 CLEAR:0 LOAD:0 PHI-1:0 TC-1:0 IOC-IN:1 PLSR-IN:0
     | IOC-BUSS-RD:1 PHI-2:1 TC-2:0 PLSR-OUT:X OP:1 nd13:1 nd29:1 rilr:111
     | rolr:111 rflr:111 rdlr:00 wrlr:00 cntlr:001
    .42 seconds, 23 steps, 144 events, 106 state changes
>set OP:0
>cy
  9.1| rilr:111 rolr:111 rflr:000 rdlr:00 wrlr:00 cntlr:000
  9.2| rilr:111 rolr:111 rflr:000 rdlr:00 wrlr:10 cntlr:000
  9.3| rilr:111 rolr:111 rflr:000 rdlr:00 wrlr:00 cntlr:000
  9.4| IO-LEFT:1 IO-TOP:0 CLEAR:0 LOAD:0 PHI-1:0 TC-1:0 IOC-IN:1 PLSR-IN:0
     | IOC-BUSS-RD:1 PHI-2:1 TC-2:0 PLSR-OUT:X OP:0 nd13:1 nd29:1 rilr:111
     | rolr:111 rflr:111 rdlr:00 wrlr:00 cntlr:000
    .38 seconds, 16 steps, 78 events, 56 state changes
>cy
 10.1| rilr:111 rolr:111 rflr:000 rdlr:00 wrlr:00 cntlr:000
```

```
 10.2| rilr:111 rolr:111 rflr:000 rdlr:00 wrlr:00 cntlr:000
 10.3| rilr:111 rolr:111 rflr:000 rdlr:00 wrlr:00 cntlr:000
 10.4| IO-LEFT:1 IO-TOP:0 CLEAR:0 LOAD:0 PHI-1:0 TC-1:0 IOC-IN:1 PLSR-IN:0
     | IOC-BUSS-RD:1 PHI-2:1 TC-2:0 PLSR-OUT:X OP:0 nd13:1 nd29:1 rilr:111
     | rolr:111 rflr:111 rdlr:00 wrlr:00 cntlr:000
    .33 seconds, 7 steps, 33 events, 25 state changes
>
>
[MICEMF - End of MIC File: MSR.MIC.3 ]
>exit
```

The "tracing" features of the CONVERT program are exemplified below. Most of the typing shown below was provided by the computer. The user typed only the lines starting with @ or >. The operations are described in the MOSSIM II User's Manual (Ref. MSII). The first block of printout can be related fairly easily to the schematic RegTot.02.sil via the corresponding `-node#'s` & `nd#'s` seen in both places. X,Y locations are in microns (each BB unit is 1.5 microns = 1/2 lambda).

```
@convert
Convert, Version 1
>re reg2btfixe
  160 nodes, 292 transistors
>sta -node13
#14 -node13  storage size:0 capacitance:595 x:635 y:1995
connectivity set:
  $25 n strength:0 gate:-node59  source:-node16  drain:-node13  x:632 y:1971
  $24 n strength:0 gate:-node56  source:-node15  drain:-node13  x:761 y:1971
  $23 n strength:0 gate:-node62  source:-node13  drain:-node20  x:533 y:1980
  $22 n strength:0 gate:-node58  source:vdd  drain:-node13  x:662 y:1980
  $21 n strength:0 gate:-node55  source:-node13  drain:-node22  x:791 y:1980
fanout set:
>sta -node29
#30 -node29  storage size:0 capacitance:590 x:635 y:1821
connectivity set:
  $64 n strength:0 gate:-node59  source:-node32  drain:-node29  x:632 y:1797
  $63 n strength:0 gate:-node56  source:-node31  drain:-node29  x:761 y:1797
  $62 n strength:0 gate:-node62  source:-node29  drain:-node35  x:533 y:1806
  $61 n strength:0 gate:-node58  source:vdd  drain:-node29  x:662 y:1806
  $60 n strength:0 gate:-node55  source:-node29  drain:-node37  x:791 y:1806
fanout set:
>sta $21
$21 n strength:0 gate:-node55  source:-node13  drain:-node22  x:791 y:1980
>sta io-left
#48 io-left  storage size:0 capacitance:6049 x:161 y:1659
connectivity set:
  $105 n strength:0 gate:-node50  source:io-left  drain:-node51  x:212 y:1653
  $78 n strength:0 gate:-node34  source:gnd  drain:io-left  x:251 y:1758
  $73 n strength:0 gate:-node38  source:vdd  drain:io-left  x:239 y:1782
fanout set:
>sta -node50
#53 -node50  storage size:0 capacitance:2271 x:95 y:1614
connectivity set:
  $111 d strength:0 gate:-node50  source:vdd  drain:-node50  x:71 y:1623
  $104 n strength:0 gate:-node52  source:-node50  drain:gnd  x:242 y:1659
fanout set:
  $111 d strength:0 gate:-node50  source:vdd  drain:-node50  x:71 y:1623
  $106 n strength:0 gate:-node50  source:gnd  drain:-node48  x:125 y:1653
  $105 n strength:0 gate:-node50  source:io-left  drain:-node51  x:212 y:1653
  $103 n strength:0 gate:-node50  source:gnd  drain:-node45  x:134 y:1662
  $93 n strength:0 gate:-node50  source:-node38  drain:gnd  x:215 y:1689
  $84 n strength:0 gate:-node50  source:gnd  drain:-node34  x:242 y:1731
>q
```

A Brief Look at Another Bristle Blocks Design

A BB design segment that shows some use of equations with control signals will be discussed. The design statement is below. The ***four lines in bold italic print*** are to be explored a little. Very little verbal discussion is presented. The presentation is mostly in the form of the annotated artwork, schematic and timing diagram. These may be studied by interested readers.

The design BB4 is just a small part of an intended multi-function Digital Clock. The section of interest here was to take the 60 cycle line as a "squared-up" digital input and derive a once per 60 second control signal that was to drive the basic clock operation. The CONTROL-TO-DATA-AND-BACK element was reviewed in the BB document and thought to be capable of generating the desired result. The documentation left an uncertain impression that was checked when MOSSIM II became available. The MOSSIM II simulation showed that the BB implementation would not work. The simulation was driven almost directly from the CIF output of BB (in a fashion similar to what was presented earlier). No details of the simulation procedure are presented.

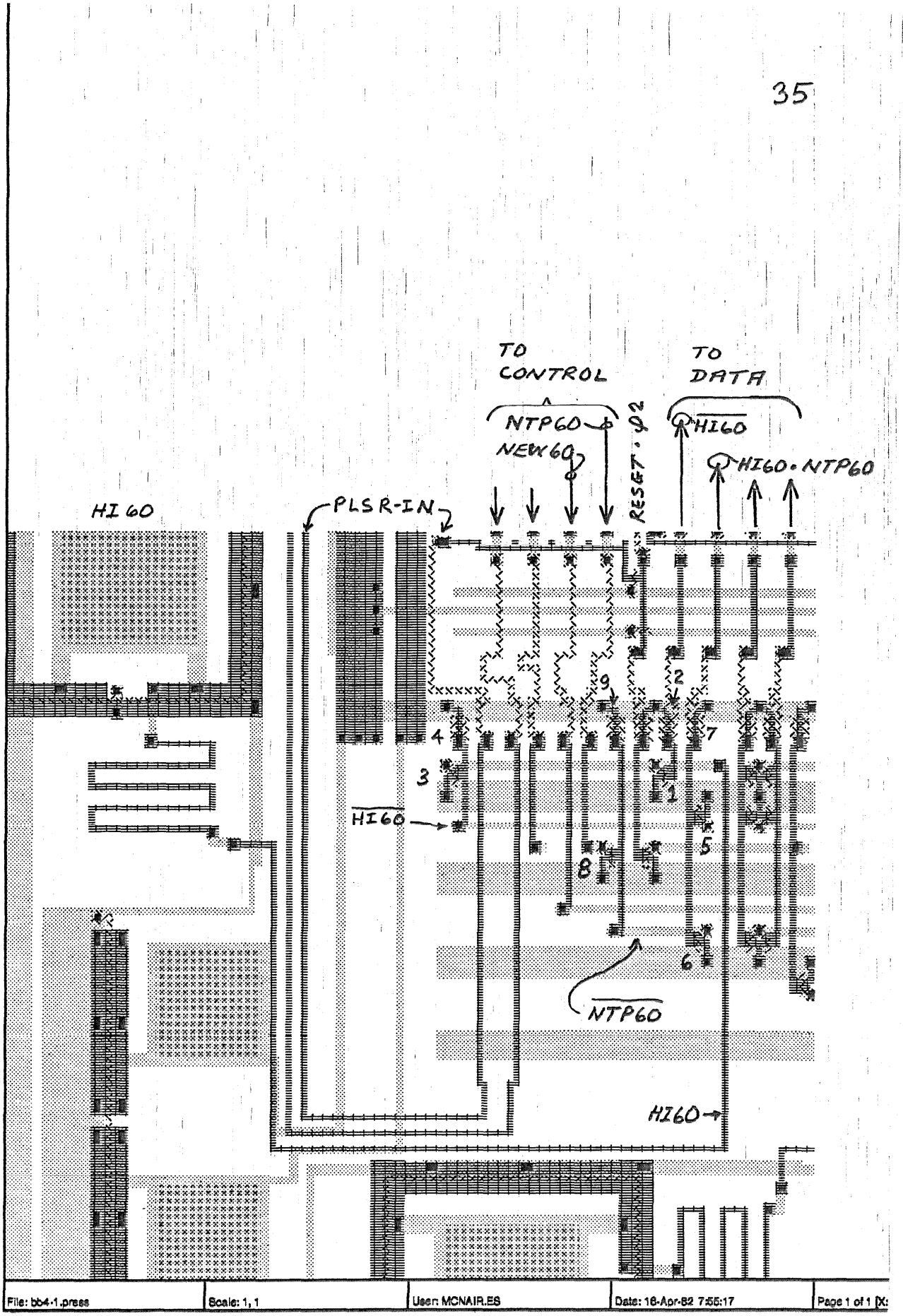
The annotated materials are presented before the discussion continues.

The Design Statement follows:

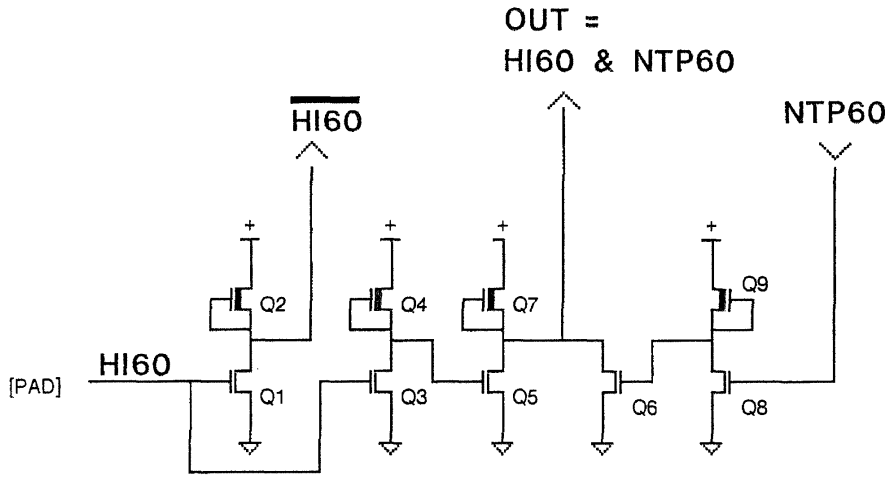
```
NAME BB4 4;   "DICKM  Dick McNair's 60 cycle to seconds converter"

FIELD RESET<1>,HI60<2>,NTP60<3>,NEW60<4>,NEWSEC<5>,U609<6>,T605<7> ;

MACRO EQ59 (CNTRID)
" This macro tests for the 59 condition (tens=5 and units=9) of "
" the counter specified by CNTRID "
   % T?CNTRID?5=I AND U?CNTRID?9=I %

PRECHARGE-BOTH PCHRG;

CONTROL-TO-DATA-AND-BACK NEW60-SEC

   REGISTER: [ SUGGEST: RESET=I, VALUE:1000,
               REFRESH: ALWAYS
             ] ,

   TO-CONTROL: {1=>NTP60; 2=>NEW60; 3=>NEWSEC; 4=>PAD } ,

   TO-DATA: {
            HI60=0                              => 1 ;  signal traced
            HI60=I AND NTP60=I                  => 2 ;  Equation traced

            U609=I AND T605=I AND NEW60=I       => 3 ;
            U609=I AND T605=I AND NEW60=I       => 4
           } ,

   LATCH: ALWAYS;
```

```
LOWER-ROM ADD1SRC
   VALUE: 000I,          ENABLE: ALWAYS;

ADDER-WITH-VALUE-CHECK U60

   INPUT-A: [SUGGEST: RESET=I OR U609=I,
             VALUE: 0000, REFRESH: NEW60=0
            ] ,
   INPUT-B: [SUGGEST: RESET=I OR U609=I,
             VALUE: 0000, REFRESH: NEW60=0,
             READ-LOWER: U609=0
            ] ,
   LOAD:  NEW60=I,

   VALUE: I00I,                  RESULT: U609  ;

ADDER-WITH-VALUE-CHECK T60

   INPUT-A: [SUGGEST: RESET=I OR /*EQ59 (60)*/ ,
             VALUE: 0000, REFRESH: NEW60=0
            ] ,
   INPUT-B: [SUGGEST: RESET=I OR /*EQ59 (60)*/ ,
             VALUE: 0000, REFRESH: NEW60=0,
             READ-LOWER: T605=0 AND U609=I
            ] ,
   LOAD:  NEW60=I,

   VALUE: 0I0I,                  RESULT: T605  ;

END
```

The next page shows the Artwork (transistors ID's with just numbers) for a decode section and the following page show the circuit schematic (transistors ID's with corresponding Q#'s) for same decoding. The circuit schematic page also presents a timing diagram. A more expanded view of the same kind of artwork as presented here was shown much earlier in the document as "Artwork: page 2". The reader may find it easier to recognize the devices in the decode example presented here if they also reference the earlier, expanded display (of similar but different logic).



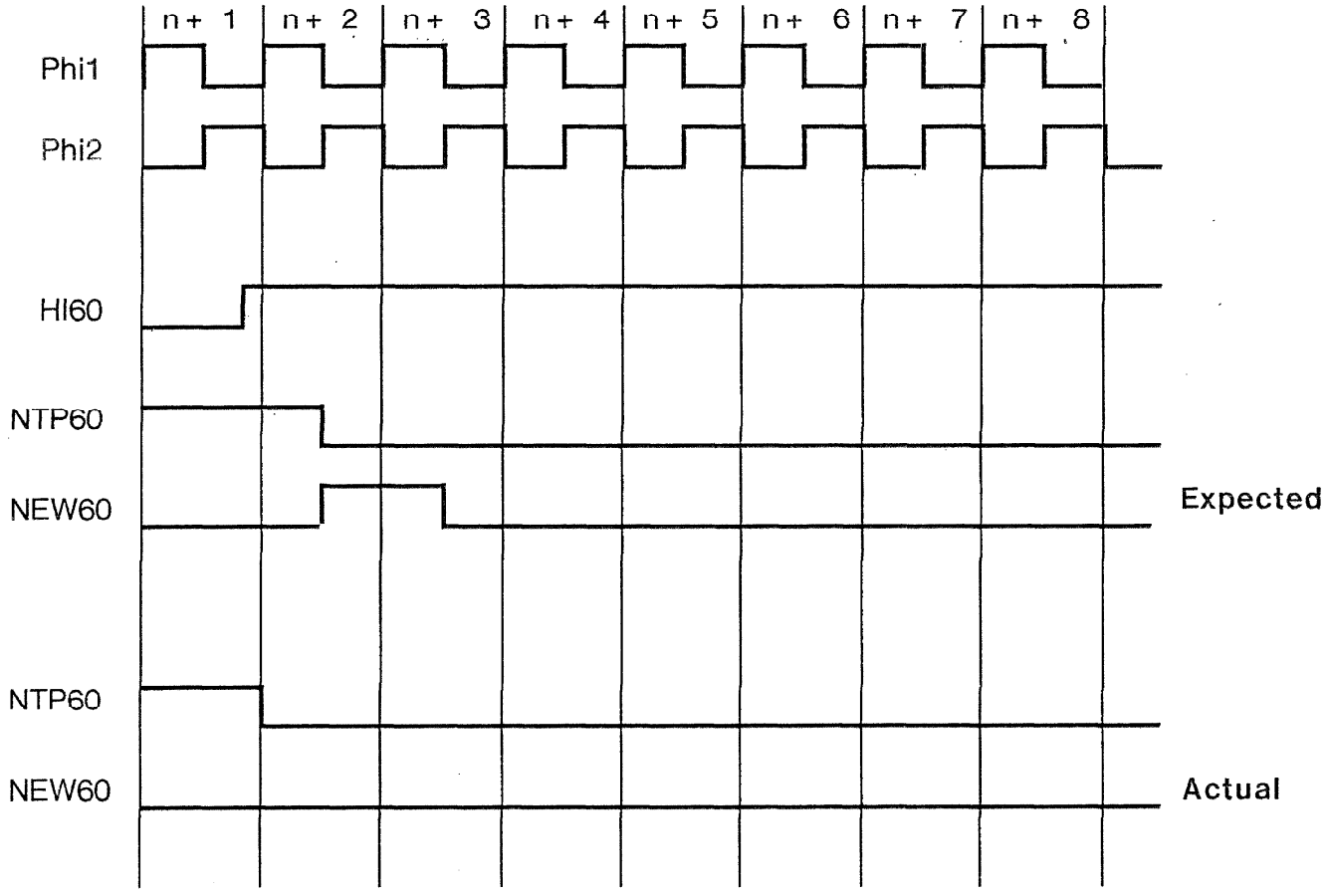
Equation Example - Decode Portion of BB4 Design Artwork



Schematic and Timing Diagram for BristleBlocks

EQUATIONS and Controls Discussion

$$OUT = \overline{\overline{HI60} + \overline{NTP60}} = HI60 \;\&\; NTP60$$



The circuit tracing shows how the "Weinburger NOR" structure was used to generate an AND equation. It also showed some inefficiency in that the `not(HI60)` that was generated as an output was regenerated for the NOR gate rather than being directly used as both an output and a NOR term.

The `CONTROL-TO-DATA-AND-BACK` element was thought to operate a a state machine in the following sense. It was expected that the signal conditions set in one clock would be usable as controls in the next clock. The simulation showed that the element directly responded to the input change and offered no isolation of "previous state" from "next state". The circuitry of the `CONTROL-TO-DATA-AND-BACK` was traced as the simulation "surprise" was studied. The schematic of the element is not presented. The timing diagram presents both the expected performance and the actual performance.

The fact that the `CONTROL-TO-DATA-AND-BACK` element does not operate a a state machine may be of interest to those considering a variety of applications for BB. It was one of the main reasons that the Digital Clock design was dropped as McNair's BB project. Some other major reasons were a poor aspect ratio of the design, a full chip before the entire function was completed and the many minor but fatal error in the current BB implementation.

References

Ref. BB "Silicon Compilation" by Dave Johannsen
(Ph. D. thesis and program documentation for Bristle Blocks)
Technical Report #4530, 1981
Department of Computer Science, California Institute of Technology
Pasadena, California

Ref. CDS "The Cell Design System" by D. Franco and L. Reed
18th Design Automation Conference Proceedings (ACM and IEEE)
1981, paper 12.4, pp. 240-247

Ref. EXT "CIF EXTRACT Help File" by Tom Hedges
CalTech and SSP Program Documentation
1982

Ref. M&C "Introduction to VLSI Systems" by Caver Mead and Lynn Conway
Addison-Wesley Publishing Company, 1980

Ref. MSII "A Switch-Level Simulator for Integrated Logic Circuits" by Randal Bryant
(Ph. D. thesis and Technical Report MIT/LCS/TR-259, March 1981)
Laboratory for Computer Science, Massachusetts Institute of Technology
Cambridge, Massachusetts 02139

"MOSSIM II : A Switch-Level Simulator for MOS LSI
User's Manual"
by Randal Bryant, Mike Schuster, Doug Whiting
16 March 1982
Department of Computer Science, California Institute of Technology
Pasadena, California

Ref. SIL "SIL A Simple Illustrator for CAP" by Chuck Thacker
to appear in Handbook of Electrical and Computer Engineering,
Sheldon Chang, ed., John Wiley & Sons, Inc.

Appendix A
Bristle Blocks
Data Path Elements

(49) ACCUMULATOR-WITH-VALUE-CHECK
(7) ADDER
(45) ADDER-WITH-VALUE-CHECK
(39) ADDING-PORT
(9) ALU
(10) ALU-WITH-FLAGS
(11) ALU-WITH-FULL-FLAGS
(21) BARREL-SHIFTER
(27) BUS-CAM
(28) CAM
(3) CONTROL-TO-DATA
(4) CONTROL-TO-DATA-AND-BACK
(2) DATA-TO-CONTROL
(6) DECREMENTER
(48) DECREMENTER-WITH-VALUE-CHECK
(33) DECREMENTING-IR
(38) DECREMENTING-PORT
(29) FUNCTION-BLOCK
(5) INCREMENTER
(52) INCREMENTER-DECREMENTER
(47) INCREMENTER-WITH-VALUE-CHECK
(32) INCREMENTING-IR
(37) INCREMENTING-PORT
(36) INPUT-IR
(12) INPUT-PORT
(14) IO-PORT
(30) LEFT-RIGHT-SHIFT
(15) LOWER-ROM
(20) MASKED-SHIFTER
(13) OUTPUT-PORT
(25) PRECHARGE-AND-BREAK-LOWER
(26) PRECHARGE-AND-BREAK-UPPER
(24) PRECHARGE-BOTH
(22) PRECHARGE-LOWER
(23) PRECHARGE-UPPER
(1) REGISTER
(17) ROM
(18) ROM-PAIR
(50) SHIFTER-WITH-VALUE-CHECK
(51) SHIFTING-ACCUMULATOR
(34) SHIFTING-IR
(19) SIMPLE-SHIFTER
(31) STACK
(8) SUBTRACTER
(46) SUBTRACTER-WITH-VALUE-CHECK
(44) SWAPPING-DECREMENTER
(43) SWAPPING-INCREMENTER
(42) SWAPPING-INPUT-PORT
(35) SWAPPING-IR
(40) SWAPPING-OUTPUT-PORT
(41) SWAPPING-REGISTERS
(16) UPPER-ROM

Appendix B

Proportional Calculations to Determine Coordinates for Signal Name Entries to the CIF File

The concept of proportional calculations relating BB units to the scaling of a composite plot are explained as follows:

$$\frac{D^{max}_{BB}}{D^{max}_{IN}} \text{ proportional to } \frac{D^{pt}_{BB}}{D^{pt}_{IN}}.$$

where:

D is a distance
BB means distance in BB units
IN means distance in inches (scaled on plot)
max means maximum distance (from lower left (origin) to upper right)
pt means a specific point

One may scale the plot in "Artwork Figure: Page One" as follows:

X_{max} = 5.8 inches

Y_{max} = 10.38 inches

The first line of the CIF file for REG2bt shows:

(symbol name: REG2BT mbb: 0#0 934#1680)

This gives the max BB units for chip, X = 934, Y = 1680.

Measurement on the plot to the center of the PHI-1 pad shows (the pad is a big target and accuracy is not extremely important):
X = 0.9 inches and Y = 2.41 inches

working with the proportion equation above, one can determine,

$$D^{pt}_{BB} = D^{max}_{BB} * D^{pt}_{IN} / D^{max}_{IN}$$

$$Y_{PHI\text{-}1} = 1680 * 2.41 / 10.38 = 390$$

$$X_{PHI\text{-}1} = 934 * .9 / 5.8 = 145$$